GENE-BASED ASSOCIATION TESTING OF DICHOTOMOUS TRAITS USING GENERALIZED FUNCTIONAL LINEAR MIXED MODELS FOR FAMILY DATA

by

Yingda Jiang

Master of Biotechnology, University of Pennsylvania, 2008

Bachelor of Science, Fudan University, China, 2006

Submitted to the Graduate Faculty of

the Graduate School of Public Health in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2015

UNIVERSITY OF PITTSBURGH

GRADUATE SCHOOL OF PUBLIC HEALTH

This dissertation was presented

by

Yingda Jiang

It was defended on

July 27th, 2015

and approved by

Daniel E. Weeks, PhD, Professor, Departments of Human Genetics and Biostatistics, Graduate School of Public Health, University of Pittsburgh

Wei Chen, PhD, Assistant Professor, Division of Pulmonary Medicine, Allergy and Immunology, Department of Pediatrics, Children's Hospital of Pittsburgh of UPMC

George C. Tseng, ScD, Professor, Department of Biostatistics, Graduate School of Public Health, University of Pittsburgh

Leming Zhou, PhD, DSc, Assistant Professor, Department of Health Information Management, School of Health and Rehabilitation Sciences, University of Pittsburgh

Dissertation Director: Daniel E. Weeks, PhD, Professor, Departments of Human Genetics and Biostatistics, Graduate School of Public Health, University of Pittsburgh

GENE-BASED ASSOCIATION TESTING OF DICHOTOMOUS TRAITS USING GENERALIZED FUNCTIONAL LINEAR MIXED MODELS FOR FAMILY DATA

Yingda Jiang, PhD

University of Pittsburgh, 2015

ABSTRACT

Gene-based association testing with rare variants requires arbitrarily aggregating or collapsing the information of the rare variants in genes into a single measure. As genotyping data can be viewed as a realization of a stochastic process that varies along the chromosome, it is more natural to summarize the genetic information using the approaches of functional data analysis. In functional data analysis, discrete genotypes are fitted by a continuous curve by using a collection of smooth basis functions. Existing generalized functional linear models (FLM) have been developed for unrelated samples to test for association between a dichotomous trait and genetic variants in a gene. In most situations, these models have higher power than well-known kernel-based methods (SKAT and SKAT-O). Here we extend this approach to accommodate family-based data using the GLOGS (genome-wide logistic mixed model/score test) approach developed by Stanhope and Abney, and develop family-based generalized functional linear mixed models (GFLMMs). This involves parallel computations to integrate out a multidimensional polygenic effect. Simulation results indicate that in most scenarios our new statistics are better than other similar statistics (famSKAT or F-SKAT), but not better than the retrospective kernel and burden statistics developed by Schaid and colleagues. We also embed FLM-smoothed genotypes in the retrospective statistics, improving the power of the kernel-based approach. We illustrate the behavior of these statistics by applying them to an age-related macular degeneration (AMD) family data set,

where, as expected, we observe strong association between AMD and *CFH* and *ARMS2*, two known AMD susceptibility genes. Our proposed GFLMM provides a new tool for conducting family-based research studies in public health for complex or multifactorial diseases. The findings may improve the knowledge of existing AMD susceptibility genes and make a positive contribution to AMD treatment and prevention.

Keywords: Functional linear model (FLM), GWAS, AMD, Linkage, Association.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PREFACE

The research related to this dissertation was supported by the AMD grant, and was performed within the Departments of Biostatistics and Human Genetics, Graduate School of Public Health, University of Pittsburgh.

I would like to express my sincere thanks to Dr. Daniel E. Weeks, my advisor and committee chairperson, not only for his advising me on this Ph.D. dissertation, but also for his guidance during the past four years in my personality, scientific attitude, and career path, which I believe will be of benefit to my lifetime.

I would like to thank all my committee members, Dr. Wei Chen for his help on building the model and constructing the statistics, Dr. George C. Tseng for his suggestions on how to write a manuscript and present my work, and Dr. Leming Zhou for his advice on defining gene boundaries and validating results.

My grateful thanks also go to Dr. Ruzong Fan for his suggestions and comments on the GFLMM approach and, his R code for the beta-smooth only model.

I sincerely appreciate Dr. Qi Yan's generating and willing to share the haplotype data for our simulation study.

I also gratefully acknowledge Drs. Yvette P. Conley and Michael B. Gorin for generating the AMD exome-chip data for our real data analysis work.

Finally, I would like to give very special thanks to my parents for understanding and supporting my decision to seek international studies at the University of Pittsburgh.

1.0 INTRODUCTION

It is not uncommon nowadays to perform a genome-wide association study (GWAS) in public health research. Researchers have generated reams of genotype data to study inherited diseases. Biostatisticians, not even enjoying a twinkle of delight in data wealth and availability, are bearing more responsibilities to develop novel methodologies to better characterize and model data structures. Challenging problems may not be just a limit to theoretical analysis of those proposed methods per se. Computational feasibility poses another inevitable hurdle to high-dimensional genotype data sets including large known pedigree structures and numerous genetic variants. Seeking promising solutions to these issues calls for time and effort put in statistical genetics. Throughout the following work, we expect to develop and present new promising statistics for analyzing dichotomous genetic traits with family-based genotype data.

1.1 RESEARCH AIMS

Our research study centered on two main aims. First, we applied existing statistical methodologies to exome chip data with family relatedness to identify age-related macular degeneration (AMD) related susceptibility loci by genome-wide linkage analysis and retrospective regression. Through these analysis procedures, novel risk variants or genes reaching genome-wide significance were reported and compared with both those already published in previous studies and the ones analyzed by our proposed methods. Second, we expounded the idea of the functional data analysis by demonstrating how to build family-based generalized functional linear mixed models (GFLMMs), and apply the new models to the analysis of

pedigree data. Our newly introduced methods also included the ones where we embedded FLM-smoothed genotypes in the retrospective kernel-based and burden tests. We then applied our newly-developed statistics to real exome chip data with known pedigree structures to identify AMD susceptibility genes.

1.2 MOTIVATION

Rare genetic variants with a minor allele frequency (MAF) less than 0.05 have triggered more research interests nowadays in testing association between a disease trait and contributive genes or loci [Lee et al., 2014]. However, the data sparsity of genetic variants set a stumbling block in the construction of the gene-based statistics. Situations may become even more complicated when we are wrestling with dichotomous traits and pedigree data. The past few years have witnessed the development of kernel-based approaches and burden tests broadly used in the rare variant analysis [Wu et al., 2011, Lee et al., 2012, Ionita-Laza et al., 2013]. Despite the emergence of methods that aggregate or collapse the rare variants, these methods, however, fail to sufficiently utilize genetic position and high-order linkage disequilibrium information. In 2013 and 2014, by using the techniques of functional data analysis, Fan et al. built functional regression models to test the association between a disease trait and multiple genetic variants [Fan et al., 2013, Fan et al., 2014]. The proposed approach yielded clues to the appropriate adjustment of high-order linkage disequilibrium by involving position information about genetic variants in the association analysis. Since their studies are focused on unrelated individuals, it brings about a natural question: Can we develop a family-base generalized functional linear model (FLM), and extend the promising approach to pedigree data analysis?

In this work, we were motivated by Fan's earlier publications [Fan et al., 2014] and enhanced the idea to further extend the generalized FLM to dichotomous traits for related individuals within known pedigrees. More specifically, to adjust pedigree relatedness, we added an unobserved polygenic term into the Fan's original beta-smooth only model. After applying functional weightings, we reduced the dimension by smoothing genotypes. To con-

struct the score statistics in the presence of nuisance parameters, we had to integrate out the polygenic term and maximized the likelihood. A numerical solution conductible to the challenging multidimensional integration in our study was propelled by the idea of tractability via parallel processing. A previously discussed mixed model-based system, the genome-wide LOGistic mixed model/Score test (GLOGS) program [Stanhope and Abney, 2012], partly met our requirement, and was modified and implemented for our parameter estimation.

Besides the elaboration on the research aims and motivation mentioned above, the following sections of this chapter will give an overall review on the existing methods for rare variant analysis in related individuals, introduce the concept of the FLM approach, and briefly discuss how the model works. In addition, since we also exploited the real exome chip data related to AMD in the UCLA/Pitt family-based study, sections will be reserved to discuss the exome chip and the genetics of AMD.

1.3 RELEVANCE TO PUBLIC HEALTH

Many research studies in public health genotype collect related individuals and pedigree structure information to investigate inherited complex diseases [Ott et al., 2011]. How to build a powerful mathematical model to better handle correlated data remains a problem in statistical genetics. In our study, we built the GFLMM to handle family data by adding an unobserved polygenic term into the existing generalized FLM approach to adjust pedigree relatedness. Our proposed model provides a new tool to conduct family-based research studies in public health with a concentration on the analysis of dichotomous disease traits. Furthermore, we applied our proposed GLFMM to the real exome chip data related to AMD, and the findings may improve the knowledge of existing AMD susceptibility genes and make a positive contribution to the etiology of AMD. We also suggested some promising findings which give clues to future studies in public health for AMD treatment and prevention.

1.4 EXISTING STATISTICAL METHODOLOGIES

1.4.1 Linkage analysis

Linkage, a term describing the coinheritance of alleles in close proximity during meiosis, constitutes one of the most consequential concepts in genome science [Lobo and Shaw, 2008]. The aim of genetic linkage analysis is to determine and quantitatively measure how risk alleles within a chromosome region cosegregate with a disease trait in a given pedigree [Ott et al., 2015]. As a traditional and once predominant tool, the backbone idea behind it comes from Mendelian recombination mapping and has now been enriched by advanced and sophisticated statistical methods. Compared with the association studies later discussed in Section 1.4.2, linkage analysis talks about using physically close genetic markers on a chromosome to investigate the tendency of coinheritance, and determine the position of disease susceptibility genes and how likely they are inherited together [Pulst, 1999]. Linkage occurs to two loci if the recombination fraction between them is less than 0.5. This kind of relation can be quantitatively constructed and expressed by a likelihood ratio from a statistical view of point. For parametric linkage analysis, the most popular and widely discussed approach is the logarithm of the odds (LOD) score method. Denoted by Z by convention, LOD score is a parametric measure of the likelihood ratio [Morton, 1955], which is defined by

$$Z = \log_{10} \frac{L(\theta)}{L(\theta = 0.5)} , \tag{1.1}$$

where L denotes the likelihood function; θ is the recombinant fraction, the ratio of the number of recombinant gametes to the total number of gametes. For a human nuclear family consisting of father, mother, and their children, Equation 1.1 can be estimated by

$$Z = \log_{10} \frac{(1-\theta)^{NR}\,\theta^{R}}{0.5^{\,NR+R}} , \tag{1.2}$$

where NR and R represent the number of non-recombinant and recombinant offspring in the pedigree, respectively.

To conquer the computational difficulties related to parametric linkage analysis, several efficient algorithms have been broadly discussed to explore underlying genetic linkage during

the past decades [Kruglyak et al., 1995, Weeks and Lathrop, 1995, Kruglyak et al., 1996]. The Elston-Stewart algorithm and the Lander-Green algorithm are the two notable ones worth mentioning. Both of the two algorithms compute exact likelihoods for pedigrees. The Elston-Stewart algorithm estimates the joint probability of phenotypes for all individuals via conditional probabilities of genotype data. The computation tends to be time-consuming under a large number of variants for investigation [Elston and Stewart, 1971, Ziegler et al., 2010]. The Lander-Green algorithm establishes transitional probabilities for identity by descent (IBD) at arbitrary chromosomal locations, and estimates the likelihood conditional on IBD sharing under hidden Markov models. Its computational burden grows exponentially with the number of individuals for investigation [Lander and Green, 1987, Ziegler et al., 2010]. A notable property about the modern Lander-Green algorithm is that it is compatible with nonparametric linkage analysis. As a matter of factor, parametric linkage analysis relies heavily on a correct specification of a predefined genetic model [Clerget-Darpoux et al., 1986]. For most multifactorial diseases, it is unlikely to specify a priori true inheritance model, which may lead to invalid estimation of LOD scores. In contrast to model-based linkage analysis, nonparametric, or model-free linkage analysis estimates IBD, an allele-sharing statistic, at a locus between pairwise relatives to test whether an observed value is greater than expected [Kruglyak et al., 1996]. To meet the requirement of analyzing large numbers of single variants, a rapid computer program MERLIN was developed by Abecasis et al. through analyzing gene patterns in a binary tree with its nodes and branches representing meioses and gene transmissions, respectively [Abecasis et al., 2002]. The program is built on sparse binary trees instead of huge likelihood arrays in the traditional algorithms and the Idury-Elston algorithm, the one with the improvements on the Lander-Green algorithm in terms of simplifying transitional matrices and using founder reduction [Idury and Elston, 1997]. Via using symmetry to combine identical gene flow patterns, Abecasis et al. laid out demonstrative arguments that the MERLIN program outperforms Genehunter or Allegro in timing and marker handling capacity [Abecasis et al., 2002]. In view of this, for our applied study in Chapter 2, we take advantage of the efficiency of the MERLIN program in handling large number of markers genotyped for pedigrees, and conduct nonparametric linkage analysis to investigate any intriguing linkage signals.

Although linkage analysis may be complained about its power and high requirement in collecting large pedigrees enriched by affected cases, it does not loose its allure in locating the chromosome segments that carry disease susceptibility genes [Pulst, 1999]. Genome-wide linkage study has been serving as a successful paradigm for identifying risk variants in Mendelian genetic disorders. To name a few for AMD for instance, at the beginning of this century, several AMD candidate genes were examined via targeted genotyping of genetic markers, which played as a prelude to genome-wide linkage analysis [Weeks et al., 2000]. The signals discovered on chromosomes 1q and 10q revealed the most two statistically significant regions amongst the early AMD studies [Klein et al., 2005, Jakobsdottir et al., 2005, Rivera et al., 2005]. These evidence provide us with genuine and reliable hits for comparison to guide later linkage or association studies in later investigations and discussions.

1.4.2 Association studies and retrospective regression techniques

It is a matter of fact that powerful genetic linkage mapping requires collecting data from related individuals and investigating the cosegregation of genetic variants and trait of interest. However, not all of the linkage studies end up with exciting results. Biostatisticians are acutely aware of some disadvantages or restrictions in terms of limited sample size to detect genes of modest effect size, low power, result interpretation, and genotype errors [Dawn Teare and Barrett, 2005]. In the purpose of gathering and analyzing more abundant population-based family data, biostatisticians turn their attention to association studies. The involvement of the unrelated individuals makes large samples possible to increase study power and investigate those genes with modest effect size. Unlike linkage analysis, association studies compare allele frequencies between disease cases and controls [Cordell and Clayton, 2005]. In drawing the inference on significant association signals, statistical analyses play a pivotal role. Traditional analysis methodologies put more weights on single variants via capturing common genetic variants independently based on the common disease common variant hypothesis [Gibson, 2011]. The restriction of these approaches lies in the ignorance of the influence of those rare variant with modest effect size on a disease trait.

With the advanced genotyping technologies gradually stepping into the mainstream, large-scale genotypic data are becoming accessible. A rise in GWASs has been found since 2005 [Hirschhorn and Daly, 2005], meanwhile facilitating the necessity in the analysis of numerous functional genetic variants with rare allele frequencies less than 0.05 [Gibson, 2011, Visscher et al., 2012]. Due to data sparsity, rare variants have to be combined into groups for association analyses, which may marginalize the outdated statistical methodologies only aimed at common variants [Lee et al., 2014]. To solve the issue by establishing the grouping, biostatisticians have been seeking the path to the identification of disease susceptibility genes by developing several modern statistical approaches. These strategies generally encompass kernel-based approach and burden tests [Drichel et al., 2014]. The main idea with kernel-based approach is to aggregate the association between genetic variants and disease traits via a kernel matrix with the structure specifying the pairwise genetic similarity between two individuals while treating covariates as fixed factors and genetic effect random factors [Lee et al., 2014]. In those early studies, the kernel-based approach was developed for quantitative traits, and assumed uncorrelated residuals. To extend the approach for pedigree data, Schifano et al. and Chen et al. introduced into their modeling scheme a random term to recover pedigree relatedness [Schifano et al., 2012, Chen et al., 2013]. More specifically, under the null hypothesis, they added in a variance component with random variation assumed to follow a multivariate normal distribution with mean 0 and variance-covariance matrix determined by inbreeding or kinship coefficients. Both sequence kernel association test (SKAT) and Schaids multiple genetic variant associating testing in 2013 served as two paradigm cases for population- and pedigree-based studies, respectively [Ionita-Laza et al., 2013, Schaid et al., 2013]. Compared to kernel-based approach, burden tests use a different strategy to create a variant-sum of burden as statistics for the variants within a genetic region [Lee et al., 2014]. Through the summation, rare variants are collapsed and represented by a single statistic, and can be used to test the association between a disease trait in a case-control study [Asimit and Zeggini, 2010]. Since the key idea behind burden tests is to collapse and cluster all rare variants into a single variable, a caution should be exercised that the variants collapsed by the variant-sum are assumed to share the same direction [Basu and Pan, 2011]. For burden tests, on the positive side, they facilitate the

analysis of rare variants with a low MAF by conquering the hurdle of data sparsity. On the negative side, however, the association may be neutralized with collapsing both risk and protective variants, resulting in of less power than kernel-based approach with the existence of a mixture of risk and protective variants within a gene region.

Although kernel-based approach and burden tests pave the way to the analysis of rare variants, a pivotal issue may lie in the non-random ascertainment of pedigree selection according to disease traits. Researchers tend to include more cases in pedigree selection process to assure an adequate power, which may introduce selection bias into the statistics for inference. Retrospective regression techniques find an unconventional approach to nicely get around the ascertainment sampling issue [Schaid et al., 2013]. As a consequential contributor to rare variant analysis for pedigree data, retrospective regression considers genotypes random factors, and regress them on a fixed disease trait. The kernel-based statistic proposed by Schaid et al., with retrospective regression behind it, takes a quadratic form $Q = (Y-\hat{Y})^T H(Y-\hat{Y})$, where $Y-\hat{Y}$ represents the residuals of disease traits, and H is a positive-semidefinite kernel matrix estimated under a weighted linear kernel by $H = GWW^TG^T$, where G is a genotype matrix denoting the copy numbers of minor alleles and W is a diagonal matrix with its elements specifying the weights for each corresponding variant [Schaid et al., 2013]. For the burden statistic with retrospective regression, Schaid et al. extended the single-variant based statistic initially suggested by Thornton et al. [Thornton and McPeek, 2010] to a gene-based burden one, and described on how to facilitate the estimation by using maximum likelihood estimates of expected IBD (EIBD) sharing.

In addition to resolving the ascertainment issue, the efficiency of conducting the retrospective regression is of another advantage. The well-built "pedgene" package (`http://cran.r-project.org/web/packages/pedgene/index.html`) in R [R Core Team, 2014] makes it possible for scanning whole genome within a reasonable time period. In Chapter 2, we will apply the retrospective kernel and burden tests to analyzing the real exome chip data to identify AMD risk genes. Furthermore, we will also discuss in later Chapter 3 on how to embed the FLM approach in this promising framework.

1.5 FUNCTIONAL LINEAR MODEL

As mentioned in preceding sections, kernel-based approach and burden tests have been developed during the past few years to overcome the difficulties of examining a single rare variant. Combined with retrospective regression techniques, these methods free themselves from modeling ascertainment process, and become more advanced and sophisticated. However, although these existing approaches have epitomized the paradigms in rare variants analysis, it is noteworthy that none of them employs the position information of genetic variants. This may bring about a potential weakness in a GWAS, that is, higher-order linkage disequilibrium fails to be taken into account [Slatkin, 2008]. An ensuing issue may be enlarged by the fact that the design of a modern GWAS nowadays is characterized by testing a high-density genetic markers or probes in a candidate region [Roberts et al., 2010], which would finally result in ambiguous association signals concerning whether a genetic variant is truly related to a trait or it is a false positive due to a non-random association. A direct perceived question is: Is there a better statistical methodology that enables the analysis of rare variants while appropriately accounting for high-order linkage disequilibrium? Such an ideal statistic is expected to incorporate variant positions the information which is always collected or accessible from web-based databases during GWASs. FLMs may hold the key to address this issue.

Functional data analysis originates in time series data modeling [Ullah and Finch, 2013]. The core content of this approach is to treat longitudinal measurements as a continuous curve [Müller and Stadtmüller, 2005]. Since in a GWAS genetic markers are tightly genotyped within a region of interest on a chromosome, such an idea can be borrowed to model the association between a trait and genetic variants by expanding predictor functions under some techniques to smooth and reduce the dimension of discrete data in a concentrated manner. Such sophisticated techniques previously discussed included the application of a series of orthogonal basis functions as smoothers and the approximation of the Karhuynen-Loéven expansion [Fan et al., 2014, Karhunen, 1947, Loeve, 1977]. Particular for a gene-based association study, Fan et al. presented an approach to revise the original FLM by taking a direct usage of genotype data with the purpose to estimate the genetic effect functions

[Fan et al., 2013, Luo et al., 2012]. In 2014, Fan's group extended their method to a more general form to handle a dichotomous trait [Fan et al., 2014]. These pioneer and influential studies, by focusing on population data with unrelated individuals, enable the analysis of rare variants or common variants, or a combination of the two. Specifically in mathematical expressions, given a disease trait and observed genotype data, the generalized FLM can be defined as

$$Y = g\left(\alpha + \int_0^1 X(t)\beta(t)\,dt\right) + \epsilon\ , \tag{1.3}$$

where Y is a vector denoting a univariate dependent variable, either continuous or discrete, following a distribution within the exponential family; $g(.)$ is an appropriate link function; α is a constant vector; $X(t)$ contains genotype data observed at a position t; $\beta(t)$ is a parameter function vector to be estimated for statistical inference; ϵ denotes the vector of error terms [Müller and Stadtmüller, 2005]. For a dichotomous trait under a logistic regression framework, the model can be specified as

$$p_i = \text{logit}^{-1}\left(Z_i^T\alpha + \int_0^1 X_i(t)\beta(t)\,dt\right)\ , \tag{1.4}$$

where Z is the underlying covariate matrix, and $p_i = P(Y_i = 1)$, the probability of disease. Under a generalized FLM, discrete genotypes are treated as a representing function to be modeled. In practice of identifying the disease associated genes, the idea is no longer generating a kernel matrix or collapsing the rare variants in a biological region, but viewing genotypes as a stochastic process that varies along the chromosome [Fan et al., 2013, Ross, 1996]. The necessity to involve functional data analysis rests with the natural functionality in the objectives of analysis and statistical modeling process itself [Ramsay and Dalzell, 1991]. Particularly for genotype data, given that position intervals for genetic variants may vary, functional data analysis may outperform any other existing approach in modeling this unbalance. Another advantage of treating discrete genotypes as functional data is the competence to adjust high-order linkage disequilibrium (not just limited to pairwise linkage disequilibrium between adjacent variants), for it does not necessarily assume that the genotypes of a certain individual are independent of each other [Ullah and Finch, 2013], and all position information within a gene boundary is sufficiently used in smoothing genotype data.

According to the studies led by Fan et al., the generalized FLM was developed for modeling a disease trait in a population-based sample [Fan et al., 2014], naturally motivating an extension to a sample with relatedness. In Chapter 3 of this dissertation, we assimilate the functional data analysis into the generalized linear mixed model framework developed by Papachristou et al. [Papachristou et al., 2011] to better handle the higher-order linkage disequilibrium for sufficiently close genetic variants. Our proposed statistics integrate the beta-smooth only model propounded by Fan et al. [Fan et al., 2013], and are adapted to a dichotomous trait in a family-based association study with the inclusion of a kinship matrix to account for pedigree relatedness. Furthermore, inspired by the unique character of Schaid's approach [Schaid et al., 2013] in modeling ascertainment mentioned in preceding Section 1.4.2, we embed our modeling strategy in retrospective regression to solve the potential ascertainment issue in our samples.

1.6 A BRIEF REVIEW OF AGE-RELATED MACULAR DEGENERATION

To evaluate the performance of our proposed model in reality, we exploits real exome chip data in the UCLA/Pitt family-based study. Besides assessing the performance, we also aim to draw a clearer picture of the genetic contributions to AMD, address uncertainty remained in the susceptibility variants previously identified but still open to debate, and shed light on the statistical methodologies applicable to family-based studies in human genetics. In this section, we give a brief review of the genetics of AMD.

AMD, a progressive neurodegenerative disease, constitutes one of the primary causes of visual impairment and irreversible blindness in the elderly of western countries. The statistics in 2004 showed that a significant increase occurs in the estimated prevalence with an increase in age for individuals of European descent [Friedman et al., 2004]. In the United States, approximately 10 million of age 40 years or older developed intermediate or advanced AMD in the mid 2000s [Friedman et al., 2004]. For people older than 80 years old, more than 15% were afflicted with AMD, and the estimated percentage is expected to rise dramatically up to 65% by 2020 [Parmeggiani et al., 2012]. The soaring morbidity rate of AMD in the

developed nations, especially causing blindness or low vision amongst the people of age 40 years or older, makes AMD a spotlight in the research of complex disease.

Understanding of the clarification and the pathogenesis of a disease per se would contribute to the advances in disease treatment, control, and prevention. AMD is not an exception. AMD of early stages is associated with the presence of atrophy of the retinal pigment epithelium, thickening of the Bruch membrane, and the appearance of large drusen [Ambati et al., 2013]. In later stages, despite an existing overlap, AMD can be characterized into choroidal neovascularization (CNV) and geographic atrophy (GA), and dichotomized in pathology based on the presence of or absence of CNV [Ambati and Fowler, 2012]. Advanced AMD is associated with the degeneration of retinal photoreceptor, or CNV and retinal pigmented epithelium cells [Ambati et al., 2013, Ambati et al., 2003]. Previous studies have led to supportive evidence that the molecular pathogenesis of AMD involves parainflammation, complement system mediated pathways, and the responses of immune system [Ambati and Fowler, 2012].

It has been clearly reported that genetic factors account for approximately 40% to 71% of the variation of AMD susceptibility [Seddon et al., 2005, Montezuma et al., 2007, Seddon et al., 2009, Gorin, 2012]. Besides the unveiling of the biological mechanism of AMD, the genetic susceptibility to AMD has been broadly discussed through publications of linkage analysis and association studies, the strategies biostatisticians have already developed to identify susceptibility genetic variants [Swaroop et al., 2007, Fritsche et al., 2014].

Probably the association studies on AMD are those of the most conspicuous exemplifications in the field of statistical genetics [Katta et al., 2009]. The rapid rise in the findings of AMD susceptibility genes during the past decade has stemmed from not only the advancement in genotyping arrays but also the refinement of the statistical approaches. The identification of association between AMD and *complement factor H* (*CFH*) in 2005 marked a prelude to the subsequent GWASs aimed at AMD [Klein et al., 2005, Haines et al., 2005, Edwards et al., 2005]. This revelation also prompted biostatisticians to turn their focus toward extensive GWASs in AMD and conclude more exciting discoveries. There is general agreement that *ARMS2* on chromosome 10 [Jakobsdottir et al., 2005, Rivera et al., 2005], together with *CFH* region on chromosome 1, is another consequential AMD susceptibility

region. The subsequent findings of *C2/CFB* region on chromosome 6 [Gold et al., 2006] and *C3* region on chromosome 19 [Seddon et al., 2013] not only confirmed those promising findings from earlier linkage studies, but also shed fresh light on the indications of novel genes/loci associated with AMD. In 2013, AMD Gene Consortium identified seven new loci reaching genome-wide significance near *COL8A1/FILIP1L*, *IER3/DDR1*, *SLC16A8*, *TGFBR1*, *RAD51B*, *ADAMTS9/MIR548A2*, and *B3GALTL* by investigating more than 77,000 cases and controls [Fritsche et al., 2013]. A more recent study conducted by the International AMD Genomics Consortium has exemplified a broader analysis on a combination of common and rare variants, unveiling sixteen novel loci associated with advanced AMD trait (Fritsche et al., in preparation; submitted to *Nat. Genet.*). It cannot be denied that to date more and more AMD associated susceptibility genes/loci have been discovered. However, most of these robust findings are based on the studies with unrelated individuals under a population design, which may evoke further investigations into the AMD related genes in family-based studies.

In summary, although molecular and immunological basis of AMD has well been laid, like many other genetic disorders, the etiology of AMD is complex, and determined by multigenic susceptibility including the complement pathway, a region of chromosome 10, lipid metabolism, extracellular matrix remodeling, and angiogenesis [Fritsche et al., 2014, Black and Clark, 2015, Cascella et al., 2014, Horie-Inoue and Inoue, 2014]. Against such a backdrop, we consider it a positive contribution to public health by examining both common and rare variants across human genome in exome chip data genotyped from related individuals. In the following Chapters 2 and 3 we report risk genes robustly associated with AMD by applying retrospective regression techniques, GFLMMs, and an embedded method combining the two approaches.

1.7 EXOME CHIP ARRAYS

Although whole-genome sequencing is no longer a utopia in GWASs, it is still cost intensive and time consuming to obtain measurements for every locus in a clinical study particu-

larly with a large number of participants recruited [Visscher et al., 2012]. It is also widely acknowledged that most findings in GWASs with traditional genotyping arrays are those variants with relatively small effect size, thus not well explaining genetic variations in the heritability of a complex disease [Cirulli and Goldstein, 2010, Robinson et al., 2014]. To address the issues, researchers have shifted their interests to those rare variants, and consider to genotype with a large sample the markers with low MAF [Wagner, 2013]. An alternative genotyping strategy is therefore developed to mainly target the exomes in human genome by capturing functional coding regions [Wang et al., 2013]. Unlike traditional genotyping arrays accommodating common variants, the design of exome chip arrays enriches for relatively rare variants, low frequency coding variants, non-synonymous variants, and other protein altering variants, technically juggling the common variants in associations analyses and the rare variants in sequencing studies (Exome Chip Design - Genome Analysis Wiki: http://genome.sph.umich.edu/wiki/Exome_Chip_Design). Exome chip arrays can also be customized with the incorporation of redundant probes to target those variants of high interest for a certain disease trait. These design properties make it possible for researchers to generate dense genotype data in a genetic region of interest, when they desire to test the hypothesis that rare variants with large effect sizes may decisively account for missing heritability of complex diseases [Manolio et al., 2009].

For a complex disease like AMD, clear evidence has been shown that rare coding variants may have play a pivotal role in explaining the genetic variation [Seddon et al., 2013, Zhan et al., 2013]. With the extensive coverage of known AMD susceptibility loci, the real Genetics of Age-related Macular Degeneration (GARM) data generated from the exome chip arrays enable a fine capture of the rare variants associated with AMD, and stand a chance of finding de novo functional coding variants remaining to be identified in our studies. These potential novel variants tend to be the rare ones that might not be widely discussed in the previous publications.

2.0 GENOME-WIDE LINKAGE ANALYSIS AND RETROSPECTIVE REGRESSION TESTS

In this chapter, we first describe the pipeline for the real GARM data cleaning. Then we elaborate on the analysis procedures via genome-wide linkage analysis and retrospective kernel-based approach and burden tests.

2.1 METHODS

2.1.1 Phenotypes

We initially attempted to collect 1,379 genotyped individuals containing 506 extended families. Among these study samples, we subset the 1,070 genotyped individuals of European ancestry for our further investigations. The AMD trait we were phenotyping and modeling was called "C_65". The "C" diagnostic scheme established an inclusive and stringent definition of AMD affected status previously discussed [Weeks et al., 2000, Weeks et al., 2004]. It defined and included the patients in the following three categories with

1. those who were clearly diagnosed as having AMD based on the evidence of extensive and/or coalescent drusen, pigmentary changes and/or the presence of GA and/or CNV;
2. those who were likely to be affected, with clear pathological changes in drusen and/or pigment epithelium; and
3. those who could not be considered AMD-free for lack of sufficient medical records, or with the presence of end-stage degeneration.

The number "65" referred to the fact that an individual was a normal control if he or she was at least 65 years old at the last exam, which could ensure it would be unlikely for the controls to develop AMD during their life time. The controls were those who had no indications of any macular or other retinal pigment epithelial changes according to eye-care diagnostics.

2.1.2 Sample Quality Investigation

To check and assess the quality of samples and genotypes (see Section 2.1.4), we applied the data cleaning pipeline built in the Gene Environment Association (GENEVA) studies project (`http://www.genome.gov/27550876`) and the methodologies previously discussed by Laurie et al. [Laurie et al., 2010].

For the 1,070 genotyped individuals of European ancestry, we carried out sample quality checks by dividing each chromosome into twelve "bins", and running B allele frequency variance analysis [Alkan et al., 2011]. The anomalous chromosome-sample pairs were identified with high B allele standard deviation more than four standard deviations from the mean B allele frequency standard deviation over all samples. Males and females were computed separately for X-chromosome. We identified 24 chromosome-sample pairs with entire or partial chromosome duplication or deletions by counting the number of bins with high standard deviation exceeding nine for each pair and gender. To further determine if there were any chromosome segment deletions or duplication, we plotted the genotyping intensities against the chromosome positions. The chromosome segments of 29 additional samples were removed due to fuzzy and noisy intensities.

2.1.3 Sample Identity and Relationship Examination

To determine whether there existed any potential gender errors, we did a misannotation gender check on the discrepancy between the annotated and genetic genders. Gender-specific means of the Y-chromosome intensity and X-chromosome heterozygosity were plotted versus X-chromosome intensity, X-chromosome heterozygosity, and autosomal heterozygosity, respectively. One sample was excluded due to a zero value of X-chromosome heterozygosity.

We further explored the pedigree relationships in the PREST software (http://www.utstat.toronto.edu/sun/Software/Prest/prest3.02/index.html) by estimating pairwise kinship coefficients and IBD sharing. We flagged any paired samples with the middle p-value of the estimates of EIBD, identical by state (IBS), and adjusted IBS (AIBS) < 0.005, and considered a violation of a certain relationship within a family. We also estimated IBD sharing by using the Method of Moments (MoM) approach in PLINK (http://pngu.mgh.harvard.edu/~purcell/plink). To make the exclusion reason more persuasive, we ranked the missing call rate for each sample involved in the potential problematic relationships. A sample with a missing call rate > 0.01 was flagged for further consideration. Filtering decisions came out based on the middle PREST p-values < 0.005, the PLINK IBD estimates, and the missing call rates that 26 samples were excluded due to relationship misspecifications with 5 samples stayed on the top list of the missing call rates. We also altered two family structures from full siblings pairs to half siblings ones due to relationship misspecification.

With respect to all of the unrelated paired samples, we further investigated the connectivity to examine whether or not an unrelated individual would be related to too many others. The connectivity coefficients were computed and ranked in descending order. The top two unrelated samples with the highest connecting relatedness were excluded.

2.1.4 Genotyping and Quality Control

With 247,870 variants released by the Center for Inherited Disease Research (CIDR) at Johns Hopkins University (http://www.cidr.jhmi.edu), we first filtered out 1,794 variants with no genotyping data released, and 487 variants with annotated genotyping errors. Since nonpolymorphic variants contributed uninformatively to linkage or association analyses, we excluded 130,872 nonpolymorphic ones before any further data cleaning steps. Due to study design issues, we figured out that 401 pairs of probes were actually detecting the same position on a certain chromosome. We therefore excluded additional 401 variants associated with a lower call frequency rate.

We then checked the quality of the sample batches by calculating *missing.n1*, missing call rates per variant over all individuals and *missing.e1*, missing call rates per individual over

all variants. Next, for all variants with $missing.e1 < 0.05$, we computed *missing.n2* over all individuals. Accordingly, for all variants with $missing.n2 < 0.05$, we computed $missing.e2$ per individual over all variants. The duplicate genotypes of 12 blind samples were excluded per a higher *missing.n2*, and in total 57 variants were dropped due to $missing.n2 > 0.05$.

Hardy-Weinberg equilibrium (HWE) may have an impact on the interpretation of linkage and association signals. To identify any variants with significant deviations of HWE, we ran the PEDSTATS software package (http://csg.sph.umich.edu/abecasis/Pedstats/index.html) to test each variant that passed the quality control. We flagged 90 variants with the p-values of the HWE test $< 10^{-7}$.

2.1.5 Statistical Analyses

We did linkage analysis and association testing for statistical analyses. In our analyses, genotypes were coded as 0, 1, and 2 to count the copy numbers of minor alleles for a certain genetic variant. Missing genotypes were coded as "NA". We used the Mega2 program (https://watson.hgen.pitt.edu/docs/mega2_html/mega2.html) to convert our data into the proper formats required by different computing programs [Baron et al., 2014].

2.1.5.1 Linkage analysis We examined autosomal chromosomes for the underlying non-parametric linkage signals by using the MERLIN software package (http://csg.sph.umich.edu/abecasis/Merlin/index.html). Missing genotypes were retained in the data for the linkage analysis. To avoid time-consuming computations on large pedigrees, we subjected to the PedStr program (http://mga.bionet.nsc.ru/soft/index.html) two pedigrees with bit size exceeding 24 to reduce the family size, and split them into four sub-pedigrees [Kirichenko et al., 2009]. To detect the underlying linkage signal for each single variant, we carried out single-point linkage analysis by treating each variant independently and assuming HWE. With linkage disequilibrium modeled, multiple-point nonparametric linkage analysis was done with the exclusion of the variants violating HWE.

For the variants on X-chromosome, we ran the linkage analysis using MINX (MERLIN in X), a program built in the MERLIN package. We also figured out pseudoautosomal regions

(PARs) on X- (60,001-2,699,520 and 154,931,044-15,526,060) and Y-chromosome (10,001-2,649,520 and 59,034,050-59,363,566) (Human genome overview - Genome Reference Consortium: http://www.ncbi.nlm.nih.gov/projects/genome/assembly/grc/human). Since the genes within PARs behave like the ones on autosomes during meiosis, we analyzed them by MERLIN as we did for those autosomal variants.

2.1.5.2 Gene-based association testing We ran multiple genetic variant testing by computing kernel association statistic and burden statistic based on the methodologies proposed by Schaid et al. [Schaid et al., 2013]. To implement this, we pulled gene boundary information from the UCSC Genome Browser (https://genome.ucsc.edu), and used the "pedgene" package in R to compute the gene-based kernel and burden statistics. Gene boundaries were determined according to the reference assembly GRCH37. We did not include any variants in the upstream or downstream sequences. We took into account different transcripts of the same gene. The genes containing a single variant were excluded from the analyses in this section. For each variant, we applied the Madsen-Browning weights [Madsen and Browning, 2009], which were based on a function of MAF estimated by $1/\sqrt{\text{MAF} \times (1 - \text{MAF})}$.

Missing values constitute a practical issue in real data analysis. Although the "pedgene" package claimed the capability of finely handling missing genotypes, we recognized that out of an unbalanced design the missing genotypes would cause the inflation of p-values. There was a concern that it might be time consuming to impute the genotypes in a family-based study with a large number of genetic markers. Alternatively, we replaced the missing genotypes for a certain variant with a number sampled from $\{0, 1, 2\}$, the associated probability of which was determined by the non-missing genotype frequency.

2.2 RESULTS AND CONCLUSIONS

A total of 976 genotyped individuals of European ancestry passed the quality control procedure. To completely connect pedigrees, we had to create dummy parents and included

non-genotyped individuals who shared the same family with those 976 individuals. Our final cleaned sample data contained 2,727 samples, 1,275 with a known AMD trait (1031 cases and 244 controls). A total number of 111,547 variants were included in our study. Among them, 26,359 were common variants (MAF ≥ 0.05), and 85,188 were rare ones (MAF < 0.05). Full details of data cleaning were summarized in Table 1.

Table 1: Marker and sample failure and adjustment

Marker filter criteria	Marker lost	Sample filter criteria	Sample lost
Failed by CIDR (no genotypes)	1,794	Anomalous chromosomes	29
Annotated genotyping error	487	Gender misannotation	1
Nonpolymorphic variants	130,872	Relationship misspecification	26
Repeated measurement	401	High connecting relatedness	2
Missing call rates $> 5\%$	57	Fuzzy sample-chromosome pairs	24
Bad probes identified	106	Duplicated samples	12
Departure from HWE	90	Pedigree structure alteration ‡	2
Markers not on autosomes †	4,310		
Markers remained in the study	111,547	Samples remained in the study	976

† Markers on X-chromosome were analyzed by MINX for linkage signals.
‡ These two individuals with altered structures were kept in the analysis.

2.2.1 Linkage Signals

We performed single- and multi-point nonparametric linkage analysis in MERLIN. To model linkage disequilibrium, we set the threshold of pairwise correlation r^2 to be 0.10, such that the variants for which $r^2 > 0.10$ were clustered to be analyzed.

For single-variant linkage signals, we presented in Table 2 the top ten hits with the highest LOD scores. As a traditional rule, LOD = 3.0 is used as a genome-wide linkage significance threshold [Churchill and Doerge, 1994]. For our single-variant signals, rs2308655 within *HLA-B* on chromosome 6 had the highest single-point LOD score (2.922). Compared with the threshold of LOD = 3.0, this linkage signal failed to reach genome-wide significance. However, we still considered it a suggestive single-variant signal given that the association of AMD with *HLA-A* and *HLA-B* were previously discussed [Goverdhan et al., 2005]. In addition, we identified another promising linkage signal (LOD = 2.714) on rs86567 within

HLA-DOA related to *HLA* classes. *HLA-DOA* belongs to *HLA* class II alpha chain paralogues, and is also located on chromosome 6, close to *HLA-B*, which might suggest an interesting AMD linkage region. Besides *HLA* classes, we also observed rs3124299 on chromosome 9 with LOD = 2.709. The variant is within *COL5A1*, a gene significantly associated with central cornea thickness in European cohorts [Vitart et al., 2010]. In Table 2 we listed corresponding multi-point LOD scores for a comparison purpose. None of them reached a genome-wide linkage significance threshold.

Table 2: LOD scores of top ten variants for single-point nonparametric linkage analysis

variant	gene	chr	position (bp)	major/minor allele	MAF	single-point LOD score	multi-point LOD score
rs2308655	*HLA-B*	6	31322303	[G/C]	0.4482	2.922	2.346
rs704124	*C12orf12*	12	90826178	[A/G]	0.2735	2.831	1.345
rs3815768	*ELL2*	5	95236459	[A/G]	0.2559	2.792	0.260
rs10799445	*ZNF678*	1	227911883	[T/G]	0.2036	2.746	0.869
rs86567	*HLA-DOA*	6	32976759	[T/G]	0.4148	2.714	1.050
rs3124299	*COL5A1*	9	137619195	[T/C]	0.4281	2.709	0.127
rs4774	*CIITA*	16	11000848	[G/C]	0.2969	2.564	0.808
rs6941421	*JARID2*	6	15089151	[A/G]	0.4092	2.564	0.078
rs1387144	*BDNF*	11	27635319	[A/C]	0.4078	2.554	0.803
rs6913635	*COL11A2*	6	33118937	[A/C]	0.0579	2.467	1.605

Top ten variants for single-point nonparametric linkage analysis were listed. Corresponding reference alleles, MAF, and multi-point LOD scores were also tabulated for comparison.

For multi-point linkage signals, the most convincing evidence of linkage was observed on chromosome 6 with the highest LOD = 3.867. Even compared to a more stringent threshold LOD = 3.3 suggested by Lander and Kruglyak [Lander and Kruglyak, 1995], the linkage we suggested here still reached genome-wide significance. To clearly demonstrate the strongest linkage peaks, we plotted the LOD scores in Figure 1 for all of the variants on chromosome 6, and particularly zoomed in the target region between 30,000,000 and 33,000,000 bp. To superimpose the known AMD risk genes, we referred to the National Center for Biotechnology Information (NCBI, `http://www.ncbi.nlm.nih.gov`), and searched for all the genes related to AMD within the genome of *Homo sapiens*. We noted that the transcripts within this region included those from *C2*, *CFB*, *SKIV2L*, *ATF6B*, *FKBPL*, *NOTCH4*, *STK19*, and *TNXB*, all

of which were AMD risk genes, giving supportive evidence that the region on chromosome 6 might be significantly linked to AMD susceptibility. To better describe the linkage area, we computed the support interval threshold by considering 1 unit below the maximum LOD score, or $\text{LOD}_{\text{support}} = \text{LOD}_{\text{max}} - 1$ [Conneally et al., 1985, Edwards, 1992]. In our analysis, the largest LOD score was 3.867, which led to $\text{LOD}_{\text{support}} = 2.867$. We plotted in Figure 2 by zooming in the region that contained all the variants associated with a LOD score greater than the $\text{LOD}_{\text{support}}$. By doing this, we achieved a narrow and precise region on chromosome 6 between 31,773,821 and 32,439,508 bp. The genes within this region were aligned below the LOD score curve to illustrate the position of the transcripts related to the linkage peaks.

Besides chromosome 6, we also plotted the multi-point linkage signals for all the other autosomes and X-chromosome in Figures 3, 4, 5, and 6 with previously reported AMD risk genes superimposed. To describe the suggestive linkage signals, we added a more inclusive threshold of LOD = 2.0 to better illustrate some genetic regions of particular interest but failing to reach a genome-wide significance threshold of LOD = 3.0. We identified suggestive linkage regions on Chromosomes 5, 8, 9, and 12 with corresponding multi-point LOD scores exceeding 2.0. We presented more details in Table 3.

Table 3: Suggestive linkage signals with multi-point LOD scores $\geq$ 2.0, but $<$ 3.0

Chromosome	Start (bp)	End (bp)	Region	Peak LOD score	Nearby risk genes
5	73,932,199	75,306,838	5p15.33	2.488	
8	100,182,333	118,165,247	8p23.3-8p23.2	2.899	
9	116,856,481	118,741,214	9p24.3	2.602	*C5*, *TLR4*
12	88,440,676	88,589,256	12p13.33	2.015	

Although linkage signals on chromosomes 5, 8, 12 were discussed by Weeks et al. as early as in 2000 [Weeks et al., 2000], our results in Table 3 gave suggestive signals on these chromosomes. No AMD risk genes were previously reported within or close to these regions, which indicated that these signals might call for further investigations. For 9p14.3 reaching a suggestive threshold, we identified two AMD risk genes *TLR4* and *C5* close to it. The association between AMD and these two genes were previously discussed by Zareparsi et al. [Zareparsi et al., 2005] and Mullins et al. [Mullins et al., 2000], respectively, which strengthened the evidence of a promising linkage region we suggested on chromosome 9.

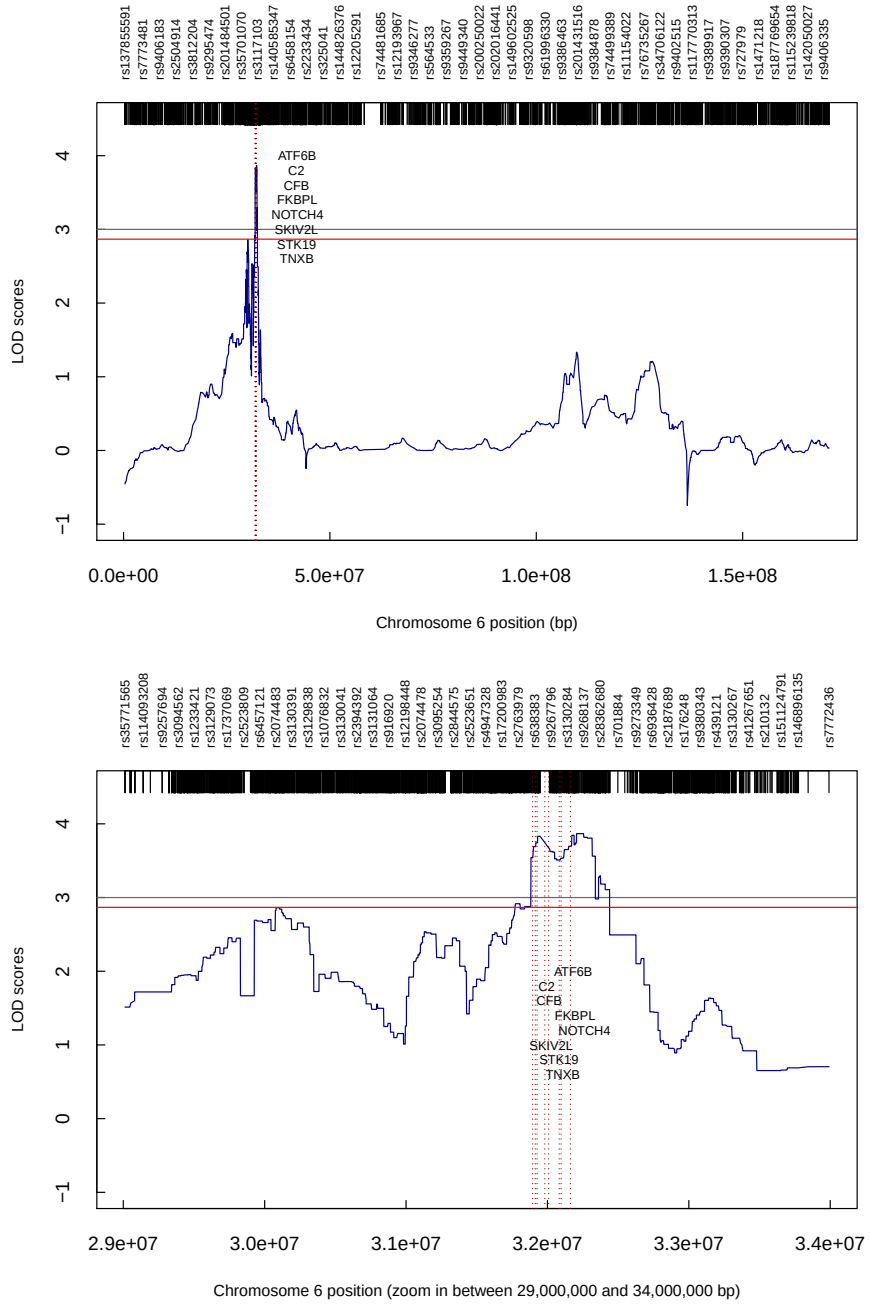

Figure 1: Linkage signals identified on chromosome 6. Linkage signals identified on chromosome 6 with a highest estimated LOD score of 3.867. The region (31,883,679-32,439,508) included the transcripts from *C2*, *CFB*, and *SKIV2L*, three AMD susceptibility genes. A threshold of LOD = 3.0 (black line) and the support interval threshold (red line drawn by 1 unit below $\text{LOD}_{\text{max}} = 3.876$) were superimposed onto the plot for comparison. The upper plot illustrated the entire chromosome 6, while the lower plot zoomed in a linkage region of interest between 30,000,000 and 33,000,000 bp.

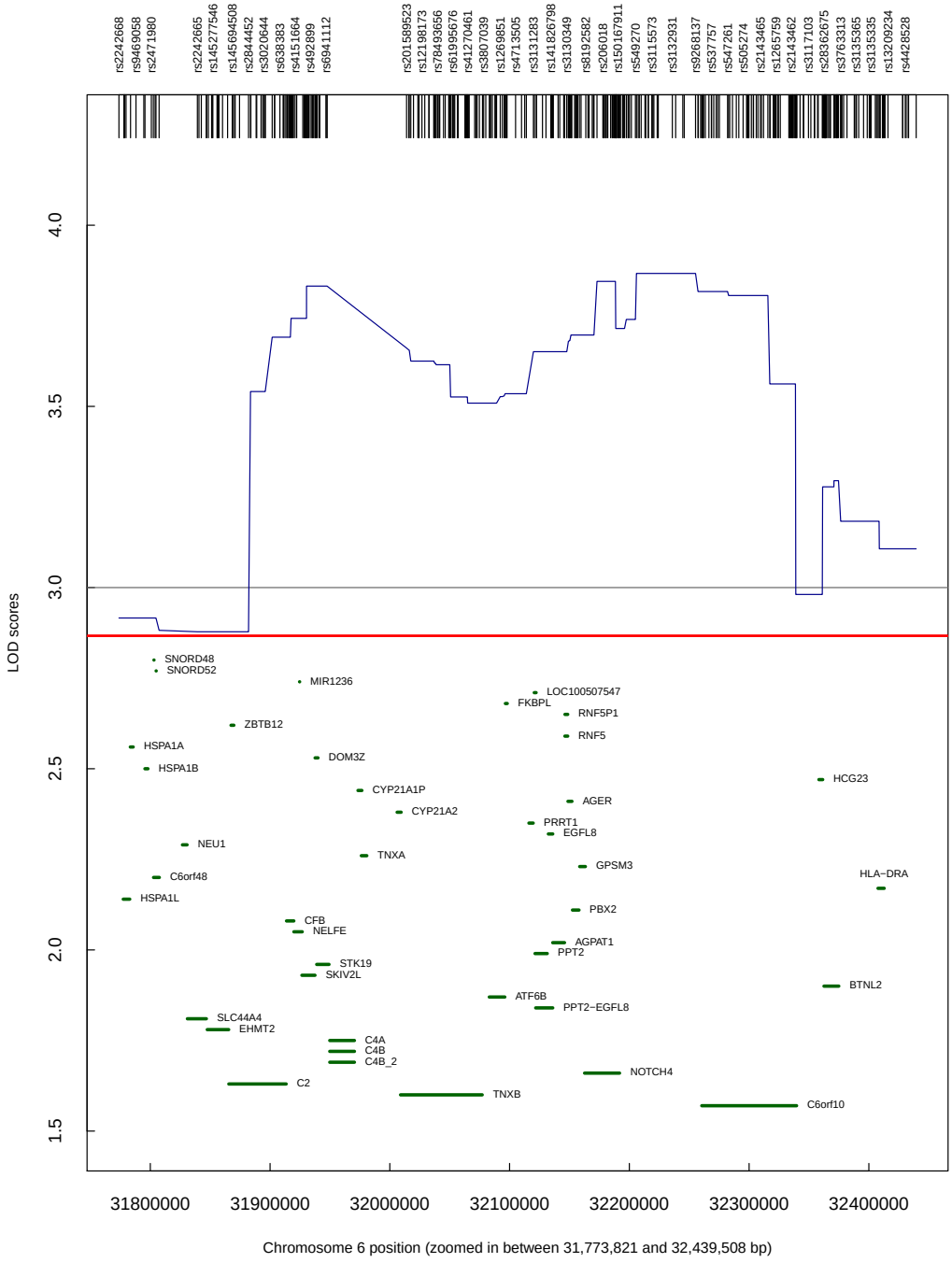

Figure 2: Linkage region identified on chromosome 6 with support interval. Linkage region identified on chromosome 6 with multi-point LOD scores > support interval threshold 2.876 (red line), 1 unit below $\text{LOD}_{\text{max}} = 3.876$. The genes within the region (31,773,821-32,439,508) were aligned below the linkage peaks. A threshold of LOD = 3.0 (black line) was superimposed onto the plot for comparison.

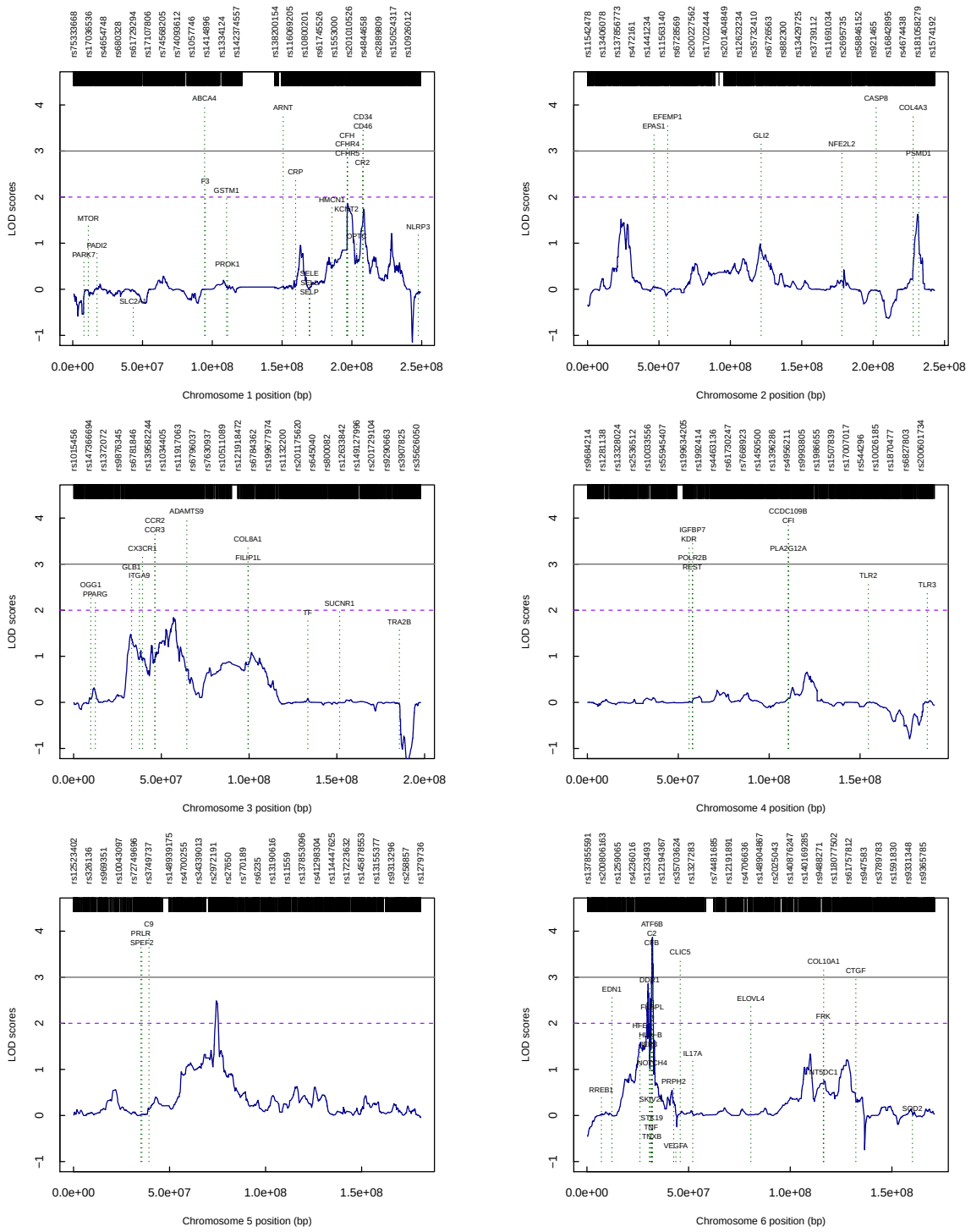

Figure 3: Multi-point LOD scores by chromosomes 1-6. Multi-point LOD scores were plotted by chromosomes 1-6. The known AMD susceptibility genes were labeled by vertical dashed lines (darkgreen). A genome-wide threshold of LOD = 3.0 (black horizontal line) and a suggestive threshold of LOD = 2.0 (pink horizontal line) were superimposed onto the plot for comparison.

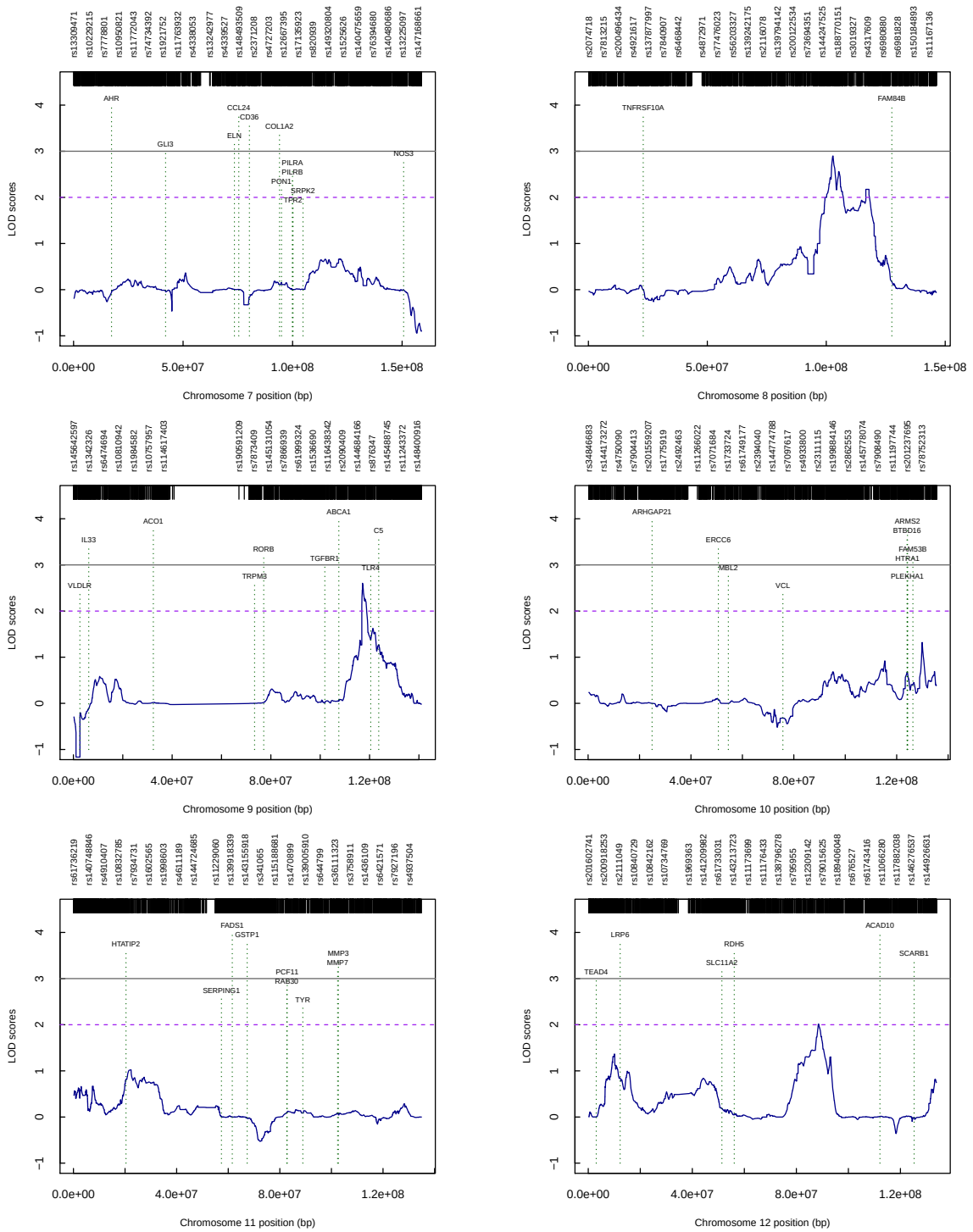

Figure 4: Multi-point LOD scores by chromosomes 7-12. Multi-point LOD scores were plotted by chromosomes 7-12. The known AMD susceptibility genes were labeled by vertical dashed lines (darkgreen). A genome-wide threshold of LOD = 3.0 (black horizontal line) and a suggestive threshold of LOD = 2.0 (pink horizontal line) were superimposed onto the plot for comparison.

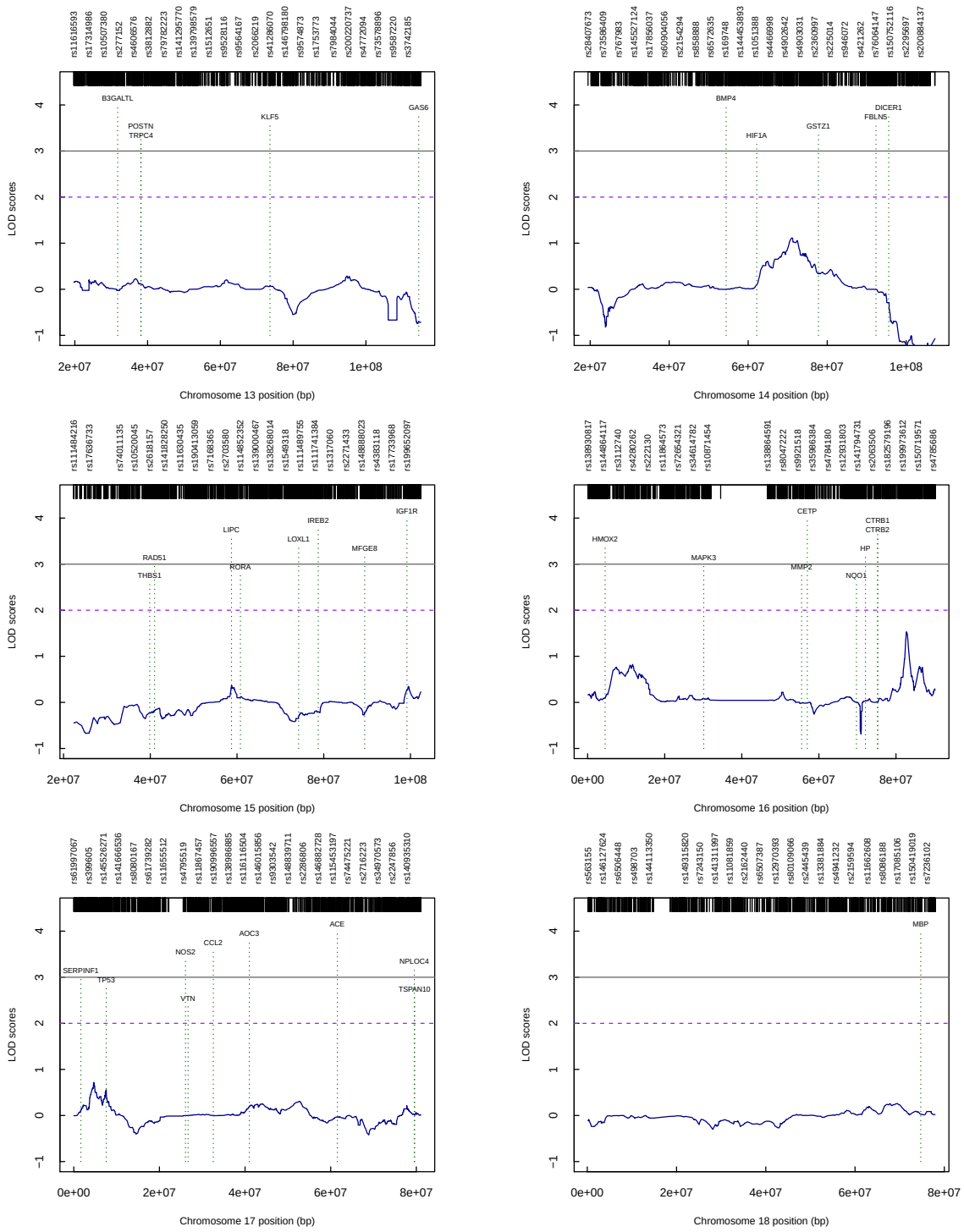

Figure 5: Multi-point LOD scores by chromosomes 13-18. Multi-point LOD scores were plotted by chromosomes 13-18. The known AMD susceptibility genes were labeled by vertical dashed lines (darkgreen). A genome-wide threshold of LOD = 3.0 (black horizontal line) and a suggestive threshold of LOD = 2.0 (pink horizontal line) were superimposed onto the plot for comparison.

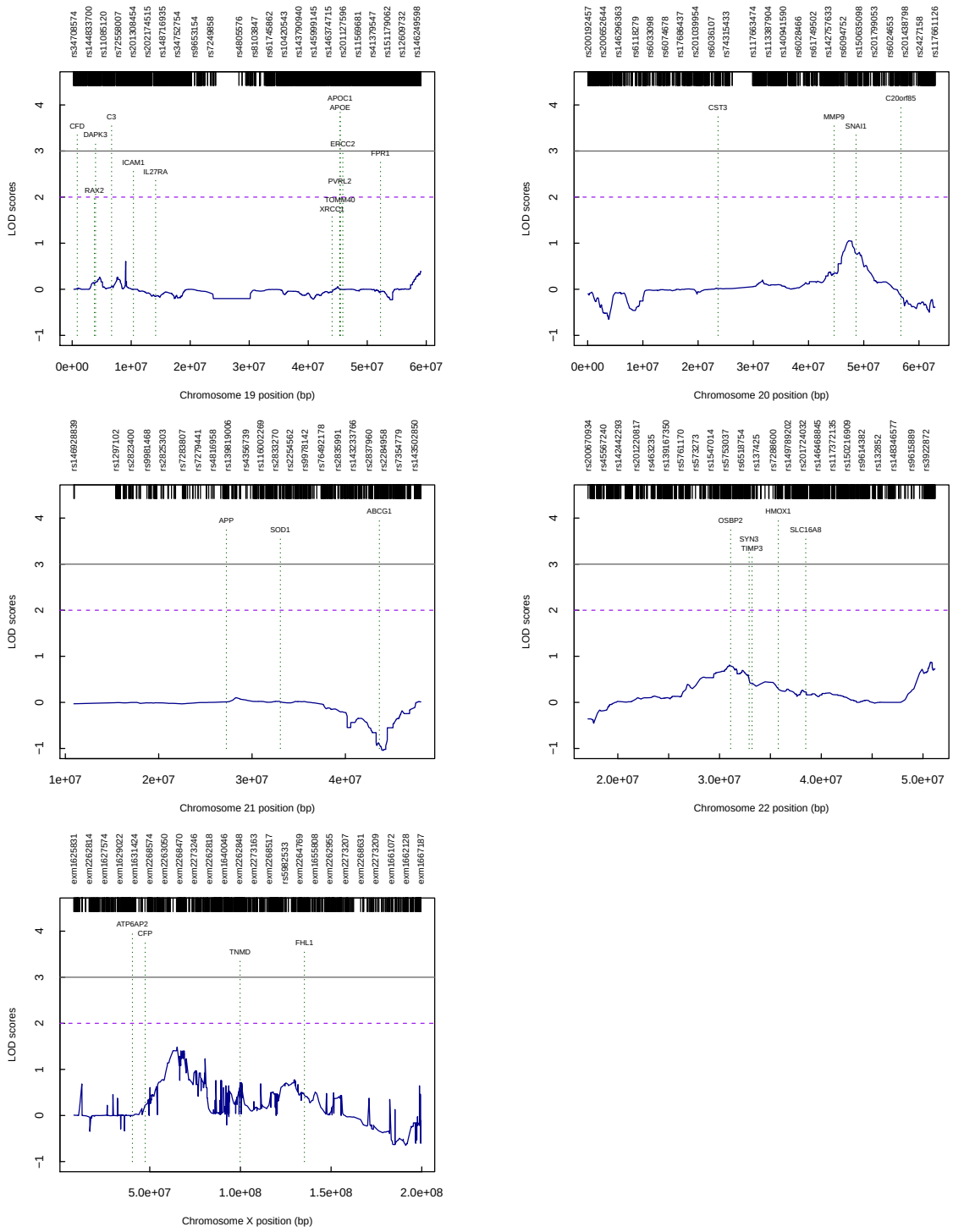

Figure 6: Multi-point LOD scores by chromosomes 19-22 and X-chromosome Multi-point LOD scores were plotted by chromosomes 19-22 and X-chromosome. The known AMD susceptibility genes were labeled by vertical dashed lines (darkgreen). A genome-wide threshold of LOD = 3.0 (black horizontal line) and a suggestive threshold of LOD = 2.0 (pink horizontal line) were superimposed onto the plot for comparison.

2.2.2 Genome-wide Association Signals

We carried out gene-based tests to investigate AMD susceptibility genes on autosomes by using retrospective kernel-based approach and burden tests. The boundaries for a certain gene might vary with the inclusion of different transcripts. Tests were conducted in two different ways. First we considered a combination of common and rare variants, and tested a total number of 16,844 autosomal genes. We controlled a genome-wide threshold of 2.97×10^{-6} after Bonferroni correction. We next excluded all common variants, and focused on the genes having at least two polymorphic rare variants. A total number of 14,849 genes were tested, associated with a threshold of 3.37×10^{-6} after Bonferroni correction. Those genes reaching genome-wide significance were summarized in Table 4.

For the analysis of the common and rare variants, we confirmed, as anticipated, that strong association was detected between AMD and *CFH* and *ARMS2*, two known AMD susceptibility genes. Both of the kernel and burden tests reached genome-wide significance (see the p-values in boldface associated with *CFH* and *ARMS2* in Table 4.

Table 4: Retrospective kernel-based approach and burden tests for a combination of common and rare variants

chr	gene	nvariant	length (kb)	start (bp)	end (bp)	kernel_MB	burden_MB
chr1	*CFH*	3	50	196621007	196670695	$\mathbf{1.537 \times 10^{-08}}$	$\mathbf{1.477 \times 10^{-07}}$
chr1	*CFH*	15	96	196621007	196716634	$\mathbf{8.253 \times 10^{-12}}$	$\mathbf{6.527 \times 10^{-10}}$
chr10	*ARMS2*	2	3	124214178	124216868	$\mathbf{6.536 \times 10^{-13}}$	$\mathbf{1.118 \times 10^{-10}}$

kernel_MB and burden_MB denote the retrospective kernel and burden tests with the Madsen-Browning weights, respectively. Genes reported with either kernel or burden, or both tests with a p-value $< 2.97 \times 10^{-6}$ (in boldface).

For *CFH*, we discovered two transcripts reaching genome-wide significance. One region of 96 kb in length (196,621,007-196,716,634) with 15 variants contained the other one (196,621,007-196,670,695) with only 3 variants, which indicated that the transcripts of *CFH* associated with AMD might be located within the smaller region between 196,621,007 and 196,670,695 bp on chromosome 1. However, for the analysis of the rare variants with MAF < 0.05, no genes reached the genome-wide significance level with either kernel-based or

burden statistics. For *ARMS2*, actually both of the two variants in our exome chip data were common variants with MAF = 0.3933 and 0.1002, respectively, so they were not included in the rare variant analysis. For *CFH*, the 96 kb region included 7 rare variants, but failed to reach the significance threshold, indicating that the common variants within *CFH* might play a pivotal role in the significant association signal. In Table 5, we listed all common variants within in *CFH* and *ARMS2*, and examined their association on an individual level by conducting the M_{QLS} test suggested by Thornton and McPeek [Thornton and McPeek, 2007].

Table 5: Single-variant association analysis for the common risk variants within *CFH* and *ARMS2*

variant	gene	chr	position (bp)	major/minor	MAF	M_{QLS}
rs800292	*CFH*	chr1	196642233	[T/C]	0.1584	6.3×10^{-06}
rs1329424	*CFH*	chr1	196646176	[A/C]	0.4578	$\mathbf{1.1 \times 10^{-12}}$
rs10737680	*CFH*	chr1	196679455	[T/G]	0.2735	$\mathbf{9.2 \times 10^{-11}}$
rs6677604	*CFH*	chr1	196686918	[A/G]	0.1279	1.7×10^{-05}
rs1410996	*CFH*	chr1	196696933	[T/C]	0.2729	$\mathbf{1.6 \times 10^{-10}}$
rs380390	*CFH*	chr1	196701051	[C/G]	0.4310	$\mathbf{2.4 \times 10^{-10}}$
rs1329428	*CFH*	chr1	196702810	[T/C]	0.2732	$\mathbf{1.8 \times 10^{-10}}$
rs1065489	*CFH*	chr1	196709774	[A/C]	0.1587	3.6×10^{-01}
rs10490924	*ARMS2*	chr10	124214448	[T/G]	0.3933	$\mathbf{5.3 \times 10^{-15}}$
rs10490923	*ARMS2*	chr10	124214251	[A/G]	0.1002	3.7×10^{-01}

M_{QLS} tests for the common risk variants within *CFH* and and *ARMS2* identified in Table 4. The M_{QLS} statistics associated with a significant p-value $< 2.09 \times 10^{-6}$ were highlighted in boldface.

In addition to *CFH* and *ARMS2*, we did not discover any other gene with a significant association signal in our exome chip data. In Chapter 3, we would analyze the same exome chip data set by using our proposed approaches for comparison.

To investigate the behavior of the kernel-based and burden tests in real data analysis, we drew quantile-quantile (Q-Q) plots of the gene-based statistics (see Figures 7 and 8). It was clear that the kernel-based statistics resulted in a higher estimated λ_{GC} value than corresponding the burden statistics, regardless of a combination of common and rare variants (1.22 vs. 1.09) or only rare variants (1.25 vs. 1.08). This interesting observation would be compared with the embedded kernel-based and burden tests discussed in Chapter 3.

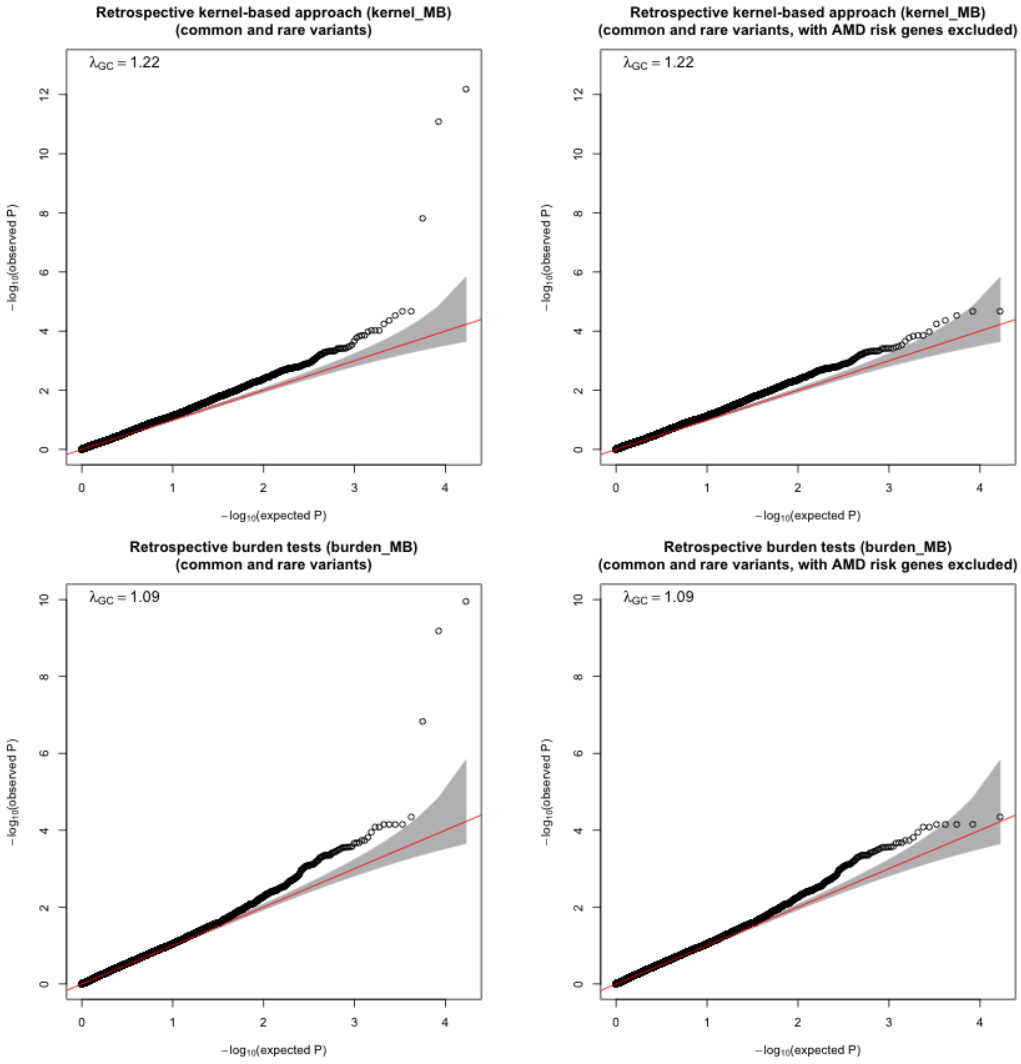

Figure 7: Q-Q plots for the retrospective kernel and burden tests with a combination of common and rare variants. Kernel and burden statistics were illustrated on the top and bottom row, respectively. Left and right column showed the genes with and without AMD risk genes, respectively. The genes contained a combination of common and rare variants. λ_{GC} values were estimated and superimposed onto the plots to denote genomic inflation factor, indicating how far the points were away from the reference line. The 95% confidence interval was shaded in gray.

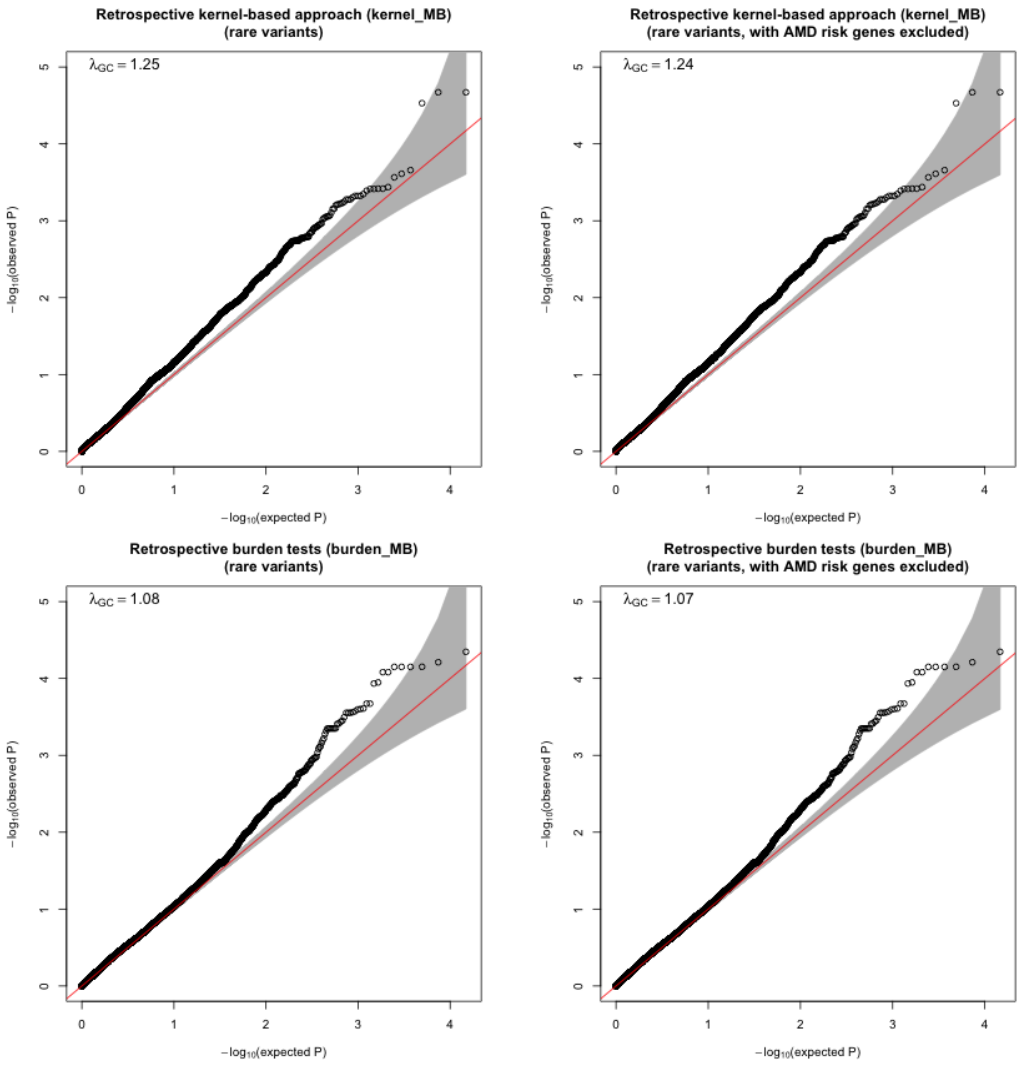

Figure 8: Q-Q plots for the retrospective kernel and burden tests with rare variants. Kernel and burden statistics were illustrated on the top and bottom row, respectively. Left and right column showed the genes with and without AMD risk genes, respectively. The genes contained only rare variants. λ_{GC} values were estimated and superimposed onto the plots to denote genomic inflation factor, indicating how far the points were away from the reference line. The 95% confidence interval was shaded in gray.

2.3 DISCUSSION

In this chapter, we conducted linkage analysis in MERLIN and gene-based association tests by using retrospective regression techniques on cleaned exome chip data. The two analyses approach the investigation of AMD etiology from different angles. Linkage analysis addresses the research question by looking at how risk alleles cosegregate and inherit in pedigrees, while association tests compare the frequency of risk alleles between cases and controls.

For the multi-point linkage analysis with linkage disequilibrium modeled, we identified a significant linkage region on chromosome 6 between 31,773,821 and 32,439,508 bp, which maps to the 6p21.31 band containing a total of 48 genes. Comparing with either a frequently used LOD = 3.0 or a more stringent threshold of LOD = 3.3, the linkage signal we suggested reached genome-wide significance with the highest LOD = 3.867. Several genes present in this linkage region were previously shown to be AMD susceptibility genes including *C2*, *CFB*, *SKIV2L*, and *HLA-DRA* belonging to *HLA* class II alpha chain paralogues. Actually, the linkage with AMD on chromosome 6 was reported and discussed in a couple of early studies. In 2003, Schick et al. observed a suggestive but less significant linkage peak on chromosome 6 [Seddon et al., 2003], but did not bound the region in detail. Later in 2007, Thompson et al. identified reported a significant region on 6q21-23 with evidence from two independent samples via investigating the rate of pigmentary abnormalities and geographic atrophy (PA/GA) [Thompson et al., 2007]. We have noted that both Schick and Thompson applied a significance threshold of p-value < 0.01, which in our opinion is less persuasive as genome-wide linkage significance. Here we provided sufficient evidence to argue that this linkage region would be a narrower band on chromosome 6 (6p21.31) with the variants contributive to the development of AMD all reaching a support interval threshold. Our findings provide one more important piece of evidence on the linkage with AMD on 6p21.

For the suggestive signals with multi-point LOD scores ≥ 2.0, we identified moderate linkage peaks on chromosomes 5, 8, 9, and 12 (see Table 3 for details). The association between AMD and 9q24 has been established by previous studies in terms of two susceptibility genes *C5* [Baas et al., 2010] and *TLR4* [Zareparsi et al., 2005], while the linkage of this region has never been clearly described. The region we discussed on chromosome 9 is

bounded between 116,856,481 and 118,741,214 bp, with multi-point LOD scores ≥ 2.0 in our analysis. According to the assembly GRCh37 and the linkage region we discussed in Section 2.2.1, *C5* is located on chromosome 9 between 123,714,613 and 123,812,554 bp, which is downstream of our suggested region. *TLR4*, also called *ARMD10* as an alias, is located between 117,704,175.and 117,717,491 bp, within the proposed region. These evidence may support the authentication of the suggestive linkage on chromosome 9. For the other suggestive linkage signals on chromosomes 5, 8, and 12, however, no previous AMD susceptibility genes were reported within or close to the corresponding region. To further investigate these suggestive linkages, we may need to collect more data, especially genotype data within these regions of interest.

For the association testing, we recognized that the missing problem in our study with an unbalanced design would inflate p-values. We addressed the missing data problem by replacing them with values derived from the frequencies of non-missing genotypes. While it might not be an optimal way to handle the missing genotypes, it would be time-consuming to do an accurate pedigree-based imputation in a genome-wide association study. Since we randomly selected a value from $\{0, 1, 2\}$ to substitute for a missing genotype, chances were that Mendelian inheritance might be violated within a pedigree. However, such violations were uncommon to occur.

Assuredly, we confirmed significant associations between AMD and *CFH* on chromosome 1 and *AMRS2* on chromosome 10 with both kernel-based approach and burden tests. In our association analysis, we considered different transcripts by varying gene boundaries for definition. An advantage by doing this is that we are able to in-depth unveil the transcripts of a gene that contribute to AMD. For *CFH*, we defined two different gene regions containing 3 and 15 variants, respectively. The smaller region (196,621,007-196,670,695) is contained in the larger region (196,621,007-196,716,634) with the same starting position. According to the results in Table 4, both of the two regions reach genome-wide significance, indicating the transcripts of *CFH* with in a small region between 196,621,007 and 196,670,695 bp on chromosome 1 may contribute to AMD. Besides examining a combination of common and rare variants with MAF < 0.05, we also investigated the effects of rare variants. When using only rare variants, no genes reached genome-wide significance. Both of rs10490923

and rs10490924 within *ARMS2* are common variants, and are excluded in the rare variant analysis. For *CFH*, the 96 kb transcript contains 7 rare variants, but is not significantly associated with AMD based on the retrospective kernel or burden tests. Through the analysis, we have demonstrated that for both *CFH* and *ARMS2*, the common variants have an impact on the significant association signals. From Table 11, not surprisingly, we have identified and confirmed these common variants within *CFH* and *ARMS2*. In fact, the individual association of these common risk variants has clearly been reported and discussed in previous publications [Hughes et al., 2006, Chen et al., 2010, Kanda et al., 2007]. Our results in Tables 4 present one more solid piece of evidence to the AMD risk loci within *CFH* and *ARMS2*. Besides the two well-known AMD risk genes, we observed no other genes reaching genome-wide significance by using the retrospective kernel or burden tests.

For the Q-Q plots in Figures 7 and 8, we observed a slight discrepancy in the estimated λ_{GC} value between kernel-based approach and burden tests. Since kernel-based approach resulted in a higher λ_{GC} value than burden tests regardless of the rarity of the variants, it might trigger an argument that the p-values for kernel-based approach were inflated in the real data analysis, or the kernel test could be more powerful than the burden test. This potential issue, if true, has not been well noted or clearly described since the development of the approach, for Schaid et al. did not further apply their proposed statistics to a real data set to conduct a GWAS. Although a GWAS is not an indispensable requirement, it may help identify some practical issues that simulation studies may not be able to identify or solve. When the AMD risk genes were removed, we did not see an obvious decrease of the λ_{GC} estimates for either the kernel-based approach or the burden tests.

In Chapter 3 (see Section 3.1.5), we introduced a novel idea that we used the B-Spline basis functions to smooth and thus reduce the dimension of genotype data, and embedded smoothed genotypes in the retrospective kernel and burden tests. Through dimension reduction, we furthered our discussions on the performance of the embedded approach and its comparison with the ordinary retrospective regression approaches. We also focused more attention on the Q-Q plots of the embedded kernel tests which would reflect the behavior of kernel-based approach in real data analysis.

3.0 GENERALIZED FUNCTIONAL LINEAR MIXED MODELS (GFLMMS) IN FAMILY DATA ANALYSIS

In this chapter, we first develop gene-based tests for a dichotomous trait and related individuals in known pedigrees by extending the generalized FLM previously discussed for population data. With the idea borrowed from the retrospective regression, we then embed the FLM-smoothed genotypes in the kernel-based approach and burden tests previously developed by Schaid et al. To assess the behavior of our newly introduced statistics, we conduct simulation studies to evaluate type I error rates and power levels, and make a comparison with some other compelling statistical methods for family-based association tests. Finally, we apply our new statistics to a real exome chip data set to identify AMD associated susceptibility genes or loci.

3.1 METHODS

3.1.1 GFLMM and Null Hypothesis for Testing

Consider a family-based genome-wide study including N participants with a dichotomous trait of interest coded as 1 and 0 denoting cases and controls, respectively. Let n denote the n^{th} individual with m genetic variants genotyped within a chromosome region. The physical locations of the m variants, denoted by $0 \leq t_1 \leq t_2 \leq \cdots \leq t_m$ in a sequence, are normalized on the unit region $[0, 1]$.

Based on the beta-smooth only model proposed by Fan et al. [Fan et al., 2013], we extend the approach to handle pedigree relatedness. Consider a logit link function on the probability

of affected status for an individual

$$\log\left(\frac{p_n}{1-p_n}\right) = \boldsymbol{\beta}^{\boldsymbol{T}} Z_n + \boldsymbol{\gamma}^{\boldsymbol{T}} \Phi(t)\, X_n(t) + \sigma a_n \ , \tag{3.1}$$

where p_n is the probability that the n^{th} individual is an affected case; $\boldsymbol{\beta}, \boldsymbol{\gamma}$, and σ are the effects of fixed covariates, genetic variants, and an unobserved polygenic term, respectively; Z_n is a design vector of fixed effects with its first element equal to 1 indicating the baseline effect; $\Phi(t)$ is a matrix containing the functions of the B-Spline or the Fourier basis; $X_n(t)$ is estimated by $X_n(t) = G_n(t)$, which takes values in $\{0, 1, 2\}$ determined by the copy numbers of minor alleles; and a_n is an unobserved random polygenic effect, following a multivariate normal distribution.

For an illustrative purpose, consider a genetic region with four variants under a series of three basis functions, Equation (3.1) can be expanded and rewritten by

$$\begin{aligned} p_n &= \text{logit}^{-1}\left\{\boldsymbol{\beta}^{\boldsymbol{T}} Z_n + \boldsymbol{\gamma}^{\boldsymbol{T}} \Phi(t)\, X_n(t) + \sigma a_n\right\} \\ &= \text{logit}^{-1}\left\{\boldsymbol{\beta}^{\boldsymbol{T}} Z_n + \begin{pmatrix}\gamma_1 & \gamma_2 & \gamma_3\end{pmatrix} \begin{pmatrix}\varphi_1(t_1) & \varphi_1(t_2) & \varphi_1(t_3) & \varphi_1(t_4) \\ \varphi_2(t_1) & \varphi_2(t_2) & \varphi_2(t_3) & \varphi_2(t_4) \\ \varphi_3(t_1) & \varphi_3(t_2) & \varphi_3(t_3) & \varphi_3(t_4)\end{pmatrix} \begin{pmatrix} g_n(t_1) \\ g_n(t_2) \\ g_n(t_3) \\ g_n(t_4)\end{pmatrix} + \sigma a_n\right\} . \end{aligned}$$

For the B-Spline basis, $\varphi_k(t) = B_k(t)$, $k = 1,\ 2,$ and 3. For the Fourier basis, $\varphi_1(t) = 1,\ \varphi_2(t) = \cos(2\pi t),\ \varphi_3(t) = \sin(2\pi t)$, with an odd number of basis functions as required. After applying the beta-smooth only smoothing, we can observe that the dimension of the genetic effect is reduced by one in this illustrative example.

By extending Stanhope and Abney's framework [Stanhope and Abney, 2012] to a more general situation, we desired to formulate a composite null hypothesis

$$H_0 : \boldsymbol{\gamma}^{\boldsymbol{T}} = \boldsymbol{\gamma}_{\mathbf{0}}^{\boldsymbol{T}} = \mathbf{0} \tag{3.2}$$

in the presence of $\boldsymbol{\beta}$ and σ as nuisance parameters.

3.1.2 Likelihood Functions

To facilitate the following presentation of the likelihoods, we let $\boldsymbol{\theta} = \{\boldsymbol{\beta}, \boldsymbol{\gamma}, \sigma\}$ and $\boldsymbol{\eta} = \{\boldsymbol{\beta}, \sigma\}$. Then the parameter space can be redefined by $\boldsymbol{\theta} = \{\boldsymbol{\gamma}, \boldsymbol{\eta}\}$.

Given the unobserved polygenic effect and for Bernoulli random variables, the conditional log likelihood function can be denoted by

$$\ell_a\left(\boldsymbol{\theta}|a\right) = \sum_{n=1}^{N} y_n \log p_n + (1 - y_n) \log\left(1 - p_n\right) \ . \tag{3.3}$$

To estimate and maximize the likelihood, a would have to be integrated out in Equation (3.3). Then the likelihood function can be expressed by

$$L(\boldsymbol{\theta}) = \int \exp\left(\ell_a\left(\boldsymbol{\theta}|a\right)\right) h(a)\, da \ , \tag{3.4}$$

where $h(a)$ is the density function of the multivariate normal distribution $MVN(0, \Sigma)$, where Σ is 2 $\times$ global kinship matrix accounting for pedigree relatedness.

Let $\hat{\boldsymbol{\theta}} = \{\boldsymbol{\gamma_0}, \hat{\boldsymbol{\eta}}\}$ denoting the constrained maximum likelihood estimation under H_0. By cubature approximation, Equation (3.4) can approximated by

$$\log L(\boldsymbol{\theta}) \approx \log\left(\sum_{c=1}^{C} \exp\left\{\ell_{\boldsymbol{a_c}}(\boldsymbol{\theta})\right\} w_c\right) \tag{3.5}$$

$$= \log\left(w * \sum_{c=1}^{C} \exp\left\{\ell_{\boldsymbol{a_c}}(\boldsymbol{\theta})\right\}\right) \ , \tag{3.6}$$

where $\boldsymbol{a_c}$ represents the multivariate normal cubature points transformed from the individual points in the Sobol cubature, and w_c is the associated cubature weights updated after each iteration. If equal weights are chosen, w no longer depends on c.

3.1.3 Derivatives and Test Statistics

The first derivative of the log likelihood function, or the score function, is a gradient vector computed by

$$\begin{aligned} S(\boldsymbol{\theta}) &= (S(\boldsymbol{\gamma}), S(\boldsymbol{\eta}))^T \\ &\equiv \left(\frac{\partial}{\partial \boldsymbol{\gamma}} \log L(\boldsymbol{\theta}), \frac{\partial}{\partial \boldsymbol{\eta}} \log L(\boldsymbol{\theta}) \right)^T \\ &\approx \frac{\partial}{\partial \boldsymbol{\theta}} \log \left(\sum_{c=1}^{C} \exp\left\{ \ell_{\boldsymbol{a_c}}(\boldsymbol{\theta}) \right\} w_c \right) \qquad \text{(by approximation)} \\ &= \frac{\sum_{c=1}^{C} \frac{\partial}{\partial \boldsymbol{\theta}} \ell_{\boldsymbol{a_c}}(\boldsymbol{\theta}) \exp\left\{ \ell_{\boldsymbol{a_c}}(\boldsymbol{\theta}) \right\} w_c}{\sum_{c=1}^{C} \exp\left\{ \ell_{\boldsymbol{a_c}}(\boldsymbol{\theta}) \right\} w_c} \ . \end{aligned} \tag{3.7}$$

With equal weights, Equation (3.7) can be reduced by canceling w

$$S(\boldsymbol{\theta}) = \frac{\sum_{c=1}^{C} \frac{\partial}{\partial \boldsymbol{\theta}} \ell_{\boldsymbol{a_c}}(\boldsymbol{\theta}) \exp\left\{ \ell_{\boldsymbol{a_c}}(\boldsymbol{\theta}) \right\}}{\sum_{c=1}^{C} \exp\left\{ \ell_{\boldsymbol{a_c}}(\boldsymbol{\theta}) \right\}} \ , \tag{3.8}$$

where

$$\frac{\partial}{\partial \boldsymbol{\theta}} \ell_{\boldsymbol{a_c}}(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - p_n)\, z_n^{\theta} \ , \tag{3.9}$$

where

$$z_n^{\theta} = (\Phi(t) G_n(t), Z_n, a_c(n)) \ . \tag{3.10}$$

Particularly, to get the score function for the genetic effect denoted by $S(\boldsymbol{\gamma})$, we set

$$z_n^{\gamma} = (\Phi(t) G_n(t))^T \tag{3.11}$$

to obtain the first derivative only with respect to $\boldsymbol{\gamma}$.

Similarly, the negative second derivative, or the observed Fisher information, is computed by

$$
\begin{aligned}
-I(\boldsymbol{\theta}) &\equiv \frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta^T}} \log L(\boldsymbol{\theta}|a) \\
&\approx \frac{\sum_{c=1}^{C} \left(\frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta^T}} \ell_{a_c}(\boldsymbol{\theta}) + \frac{\partial}{\partial\boldsymbol{\theta}} \ell_{a_c}(\boldsymbol{\theta}) \frac{\partial}{\partial\boldsymbol{\theta^T}} \ell_{a_c}(\boldsymbol{\theta}) \right) \exp\left\{\ell_{a_c}(\boldsymbol{\theta})\right\} w_c}{\sum_{c=1}^{C} \exp\left\{\ell_{a_c}(\boldsymbol{\theta})\right\} w_c} \\
&\quad - \frac{\partial}{\partial\boldsymbol{\theta}} \log\left(L(\boldsymbol{\theta})\right) \frac{\partial}{\partial\boldsymbol{\theta^T}} \log\left(L(\boldsymbol{\theta})\right) \ ,
\end{aligned} \tag{3.12}
$$

which reduces to

$$
\begin{aligned}
-I(\boldsymbol{\theta}) &\approx \frac{\sum_{c=1}^{C} \left(\frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta^T}} \ell_{a_c}(\boldsymbol{\theta}) + \frac{\partial}{\partial\boldsymbol{\theta}} \ell_{a_c}(\boldsymbol{\theta}) \frac{\partial}{\partial\boldsymbol{\theta^T}} \ell_{a_c}(\boldsymbol{\theta}) \right) \exp\left\{\ell_{a_c}(\boldsymbol{\theta})\right\}}{\sum_{c=1}^{C} \exp\left\{\ell_{a_c}(\boldsymbol{\theta})\right\}} \\
&\quad - \frac{\partial}{\partial\boldsymbol{\theta}} \log\left(L(\boldsymbol{\theta})\right) \frac{\partial}{\partial\boldsymbol{\theta^T}} \log\left(L(\boldsymbol{\theta})\right)
\end{aligned} \tag{3.13}
$$

with equal weights, where

$$
\frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta^T}} \ell_{a_c}(\boldsymbol{\theta}) = \sum_{n=1}^{N} (p_n^2 - p_n)\, z_n^{\theta}\, z_n^{\theta^T} \ . \tag{3.14}
$$

We recognize that a similar formula to Equation 3.12 was previously derived by Stanhope and Abney. However, there should be a "+" sign in the numerator of the first instead of a "−" sign as they derived in their Appendix 3 on page 9.

To construct the test statistic, we consider Rao's score test. A general form of the test statistic can be formulated by $R = S(\boldsymbol{\theta})^T I(\boldsymbol{\theta})^{-1} S(\boldsymbol{\theta})$ [Rao, 1948, Rao, 2005]. Under the null hypothesis (3.2) in the presence of the estimated nuisance parameters $\hat{\boldsymbol{\eta}}$ under the maximum likelihood,

$$
R_r = S(\boldsymbol{\gamma_0})^T I(\boldsymbol{\gamma})^{-1} S(\boldsymbol{\gamma_0})\Big|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}},\sigma=\hat{\sigma}} \sim \boldsymbol{\chi}^2_{\boldsymbol{\kappa_\gamma}} \ , \tag{3.15}
$$

where $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}$ are the maximum likelihood estimates of $\boldsymbol{\beta}$ and σ, respectively. To make inference of the genetic effect from the score statistics, chi-squared distribution function is used to convert the score statistic to desired p-value under the null hypothesis.

To compute $I(\boldsymbol{\gamma})^{-1}$, note that the elements of $I(\boldsymbol{\theta})$ are determined by the derivatives with respect to the parameters of genetic effect $\boldsymbol{\gamma}$ and the nuisance parameters $\boldsymbol{\eta}$, and $I(\boldsymbol{\gamma})^{-1}$ can be determined by the manipulation of the block matrix $I(\boldsymbol{\theta})$. Here we define

$$I(\boldsymbol{\theta}) \equiv -\frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^{\boldsymbol{T}}} \log L(\boldsymbol{\theta}) = \begin{bmatrix} I_{\gamma\gamma}(\boldsymbol{\theta}) & I_{\gamma\eta}(\boldsymbol{\theta}) \\ I_{\eta\gamma}(\boldsymbol{\theta}) & I_{\eta\eta}(\boldsymbol{\theta}) \end{bmatrix} . \tag{3.16}$$

Then

$$I(\boldsymbol{\gamma})^{-1} = \left[I_{\gamma\gamma}(\boldsymbol{\theta}) - I_{\gamma\eta}(\boldsymbol{\theta}) I_{\eta\eta}(\boldsymbol{\theta})^{-1} I_{\eta\gamma}(\boldsymbol{\theta}) \right]^{-1} . \tag{3.17}$$

3.1.4 Computational Issues in Multidimensional Integration

As mentioned in Section 3.1.2, to estimate the maximum likelihood we have to integrate out the unobserved polygenic effect. This integration is a multidimensional one, and its computational hurdle and efficiency may depend on the sample size and the complexity of pedigree structures. To conquer the technical challenges, we resorted to the GLOGS program (http://www.bioinformatics.org/~stanhope/GLOGS), and addressed the high-dimensional integration via a tractable and parallel implementation [Stanhope and Abney, 2012].

As previously discussed in detail, the GLOGS program features the estimation of parameters from a logistic regression model with a random polygenic effect term and the performance of score tests between a dichotomous disease trait and a single marker. In our study, we took full advantage of the first characteristic to achieve the aim of integration. However, rather than simply applying the relevant programs in the GLOGS package, we considered several modifications before adopting it to our work. The updates are summarized in Table 6 and discussed in detail in the following presentation.

3.1.4.1 The Gauss-Newton algorithm with step-halving The original GLOGS program uses the Gauss-Newton algorithm to update the parameter space by updating $\hat{\boldsymbol{\Theta}}^{i+1} = \hat{\boldsymbol{\Theta}}^{i} - D^2_{\boldsymbol{\Theta}} \log L^i(\hat{\boldsymbol{\Theta}}^i)^{-1} D_{\boldsymbol{\Theta}} \log L^i(\hat{\boldsymbol{\Theta}}^i)$, where $\hat{\boldsymbol{\Theta}}^i$ is the estimated parameter space update at the i^{th} iteration, and $D_{\boldsymbol{\Theta}}$ and $D^2_{\boldsymbol{\Theta}}$ denote the first and second derivative with respect to the parameter space $\boldsymbol{\Theta}$, respectively. Despite being a broadly accepted and applied technique,

Table 6: Summary of the updates on the GLOGS program

	Our Modifications	Original GLOGS Program
Estimation Algorithm	Gauss-Newton with step-halving	Gauss-Newton
Stopping Criteria	Likelihood comparison	Jacobian matrix comparison
Cubature Weights	Equal weights	Bayesian updating weights

it may result in the "downhill" (more negative in our case) estimations of the log likelihood in a few computational iterations. Since our aim is always maximizing the log likelihood, the algorithm without any modifications may be deviate and inefficient to achieve the maximization purpose. To improve the algorithm, we instead used the Gauss-Newton updates with a step-halving process to ensure a consistent uphill estimation of the log likelihood in each iteration by updating

$$\hat{\mathbf{\Theta}}^{i+1} = \hat{\mathbf{\Theta}}^{i} - \alpha * D^2_{\mathbf{\Theta}} \log L^i(\hat{\mathbf{\Theta}}^i)^{-1} D_{\mathbf{\Theta}} \log L^i(\hat{\mathbf{\Theta}}^i) \ , \tag{3.18}$$

where $\alpha = (1/2)^n$, and n is a positive integer starting from 1. We compare the log likelihood in the $(i+1)^{th}$ iteration with the one in the i^{th} iteration to determined whether or not the step-halving needs to be invoked by applying an *if* condition loop:

if $\left(\log L^i(\hat{\mathbf{\Theta}}^{i+1}) - \log L^i(\hat{\mathbf{\Theta}}^i) < -10^{-8}\right)$

{ *invoke the step-halving procedures* }

else

{ *move on to the next iteration* }

3.1.4.2 Stopping criterion The stopping criterion in the original GLOGS program deals with a comparison of the Euclidean norms of the Jacobian matrix from the two adjacent iterations: $\|D_\Theta \log(L^i(\hat{\mathbf{\Theta}}^i)\| - \|D_\Theta \log(L^i(\hat{\mathbf{\Theta}}^{i+1}))\| < \delta$, where $D_{\mathbf{\Theta}}$ denotes the first derivative with respect to the parameter space, and δ is a small number. To the best of our knowledge, such a convergence criterion is seldom used in estimating parameters as a stopping criterion.

Actually it may result in some issues. Here is a straightforward counterexample that nullifies the stopping criteria in one-dimensional space. If we define

$$f(x) = -(x-1)^2 \quad (x \in \mathbb{R}) \ , \tag{3.19}$$

it is obvious that $f(x)$ is maximized at $x = 1$, and the first derivative of $f(x)$ is

$$\frac{df(x)}{dx} = 2 - 2x \quad (x \in \mathbb{R}) \ . \tag{3.20}$$

Particularly, we let $x^i = 0.5$, $x^{i+1} = 0.5000004$, and $\delta = 10^{-6}$. By applying the stopping criteria, we will obtain $\|2 - 2x^i\| - \|2 - 2x^{i+1}\| = 8 \times 10^{-7} < \delta = 10^{-6}$. However, $x^i = 0.5$ is still far from 1, the value we anticipate to maximize $f(x)$.

To address the problem, we suggested a different but more widely-acknowledged convergence rule in our improved program by considering the relative function criterion

$$2\left|\log(L(\hat{\boldsymbol{\Theta}}^{i+1})) - \log(L(\hat{\boldsymbol{\Theta}}^{i})\right| \ \leq \text{FTOL} * \left(\left|\log(L(\hat{\boldsymbol{\Theta}}^{i+1}))\right| + \left|\log(L(\hat{\boldsymbol{\Theta}}^{i})\right| + \ \epsilon\right) \ , \tag{3.21}$$

where ϵ is a small number, and we set FTOL $= 10^{-8}$ to speed up the convergence without significantly loosing the estimation precision.

3.1.4.3 Cubature weights To approximate the multidimensional integral, the original GLOGS program computes the cubature weights using a strategy called Bayesian updating after estimating the parameter space at a certain iteration. It also reassigns the cubature point weights near zero to be zero in order to accelerate the computation speed. However, this strategy may lead to the involvement of a single weight to approximate a high-dimensional integral in a large data set with more than 1,000 samples. Although the strategy appears to be working for the sample data sets included in the GLOGS program package, we suggest using equal weights to better approximate the desired integration instead of applying the Bayesian updating. The advantages of our suggested weightings are obvious. If C cubature points were applied to the approximation, the weight for each iteration would simply be $1/C$. Another nice aspect is that if equal weights are chosen, our proposed model can be reduced by canceling out the weighting term (see Equations 3.8 and 3.13), leading to a straightforward derivation and more convenient and efficient programming.

3.1.4.4 Small sample testing We tested our modified GLOGS program in two small data sets with 7 and 25 individuals, respectively, to estimate parameters and maximum likelihoods. Then we used the "cubature" package (https://cran.r-project.org/web/packages/cubature/index.html) and the "R2Cuba" package (https://cran.r-project.org/web/packages/R2Cuba/index.html) to compute the likelihoods in R at the estimated parameter vector as returned by the GLOGS program. The results were summarized in Table 7. Compared with the results generated by the R packages, the original GLOGS program failed to approximate a likelihood value close to a true maximum likelihood estimate (see Table 7, *A* lines). After we modified the program, the results were consistent with those obtained by the existing R packages in our small testing data sets.

Table 7: Multidimensional integration in small samples

Data sets		Parameters and likelihood estimation in GLOGS			Likelihood estimation in R packages			
testing methods	sample size	genetic effect (GLOGS)	polygenic effect (GLOGS)	log likelihood$^{(1)}$ (GLOGS)	Inv_logL$^{(2)}$ (cubature)	MVN_logL$^{(3)}$ (cubature)	Inv_logL$^{(2)}$ (R2Cuba)	MVN_logL$^{(3)}$ (R2Cuba)
A	7	-18.184470	-105.546200	-0.054201	-4.619314	-4.618390	-4.614736	-4.594271
B	7	-0.916300	0.000000	-4.187887	-4.187887	-4.187408	-4.187887	-4.180207
C	7	-0.916291	0.000016	-4.187887	-4.187887	-4.187408	-4.187887	-4.180207
D	7	-0.916291	0.000016	-4.187887	-4.187887	-4.187408	-4.187887	-4.180207
A	25	-25.07313	-63.17299	-0.039499	NA†	NA†	-17.39616	-53.92567
B	25	-1.386294	0.000000	-12.51006	NA†	NA†	-12.51006	25.73193
C	25	-1.386294	0.000030	-12.51006	NA†	NA†	-12.51006	25.73193
D	25	-1.386294	0.000030	-12.51006	NA†	NA†	-12.51006	25.73193

† The R "cubature" package cannot handle the adaptive integration of a function with a high dimension.

Multi-dimensional integration in small samples by using the (modified) GLOGS program, R "cubature" package, and R "R2Cuba" package. *A*: Original GLOGS program with no modifications; *B*: Original GLOGS program with equal weights; *C*: Our improved version with equal weights and corrected convergence criterion; *D*: Our version with equal weights, corrected convergence criterion, and step-halving procedure; (1) log likelihood: Estimated maximum log likelihood; (2) Inv_logL: Maximum log likelihood estimated by using inverse transformation techniques; (3) MVN_logL: Maximum log likelihood estimated by transforming cubature points to a multivariate normal distribution with the kinship covariance matrix, and then integrating on the [0,1] hypercube.

3.1.5 Embed FLM-smoothed Genotypes in Retrospective Regression

A random sampling process may have an influence on the bias of the statistics computed in Section 3.1.3. To sufficiently utilize genetic positions while eliminating ascertainment bias, we embed the smoothed genotypes in the retrospective regression techniques, and construct kernel and burden statistics with beta-smooth retrospective analysis. To do this in an efficient manner, we subject smoothed genotypes to the "pedgene" package developed in R. For this embedded approach, we only consider a smoothing process by using the B-Spline basis. We note that not all variants within a gene region can be reduced to a fixed arbitrarily small dimension, especially for those large genes containing a high number of genetic variants within them. This is because the the copy number of the minor allele for a particular variant cannot exceed 2, which means that its genotype data must satisfy $\bar{G}/2 < 1$, where $\bar{G}$ is the mean of the genotypes for the variant across all of the individuals. To get around this issue and reduce more dimensions, we apply the smallest number of basis that can allow the analysis to run in the "pedgene" package. Since genetic variants within a gene are weighted by the B-Spline basis, we apply equal weights when computing the kernel and burden statistics.

3.1.6 Simulation Study

To evaluate the performance of our test statistics, we simulated data to estimate empirical type I error rates and power levels.

3.1.6.1 Pedigrees We first simulated 25 families by randomly choosing progeny sizes from a negative binomial distribution [Cavalli-Sforza and Bodmer, 1999]. We assumed that each child within the second generation has a 25% chance of having offspring. Pedigree connection was checked by the PEDSTATS software to ensure each individual was correctly and completely connected. The final structure of the pedigrees included 228 individuals (119 males and 109 females; 70 founders and 158 nonfounders) within 25 families. The pedigree size ranged from 4 to 24 with an average value of 9.12.

3.1.6.2 Genetic variants A pool of 10,000 haplotypes was generated by Dr. Qi Yan under the coalescent approach by using the COSI program [Schaffner et al., 2005]. The haplotypes were modeled on a European population and span a length of 200 kb on a region of chromosome 1. We considered 14 kb, a reported median of gene length [Lander et al., 2001], as a desired biological region that included 261 genetic variants. To target rare variants, we removed those variants with MAF ≥ 0.05.

To evaluate power levels, we sampled two haplotypes at random within the 14 kb region for each founder. For each nonfounder, we chose one haplotype at random from his or her parents. Genotypes were constructed by summing up two haplotypes for each individual to indicate the copy number of minor alleles.

3.1.6.3 Trait assignment and ascertainment Trait status was determined for each individual based upon the genotypes. To do this, we considered a logistic regression genetic model to compute the probability of being affected for the i^{th} individual:

$$p_i = \frac{\exp\left(\beta_0 + \sum_j \log\left(\text{OR}_j\right) g_{ij} + \sigma a_i\right)}{1 + \exp\left(\beta_0 + \sum_j \log\left(\text{OR}_j\right) g_{ij} + \sigma a_i\right)}\,, \tag{3.22}$$

where we let $\beta_0 = -2.94$ and $\sigma = 0.2$ to set the disease prevalence to 5% and to set the polygenic effect, respectively; g_{ij} is the copy number of minor alleles of the j^{th} genetic variant for the i^{th} individual; OR_j is the odds ratio associated with the j^{th} genetic variant [Larson and Schaid, 2014]. The odds ratios in the model were set in the following two ways. For constant OR models, all risk and protective variants were associated with OR = 2 and OR = 0.8, respectively. For MAF-dependent OR models, ORs were determined by $\text{OR}_j = \frac{\log c}{4} \left|\log_{10} \text{MAF}_j\right|$ as proposed by Wu et al., where we let $c = 3,\ 4$ or 10 to associate a rare variant with MAF equal to 0.0001 with OR = 3, OR = 4 or OR = 10, respectively [Wu et al., 2011]. After assigning the phenotype for each individual, we applied an ascertainment criterion to variants selection to ensure that each pedigree must contain at least 1 pair of affected siblings, either in the second or third generation, or both. Through the ascertainment, we anticipated to sample cases enriched for the dichotomous trait, thus weakening the influence of the polygenic effect.

3.1.6.4 Null genes To simulate null genes for type I error rates evaluation, we had to generate genotype data that was unrelated to the trait data. To do this while saving time and lowering the computational burden, we kept the trait variable as assigned in Section 3.1.6.3, but randomly assigned haplotype data for all the founders, transmitted the haplotypes to their offspring, and finally converted the haplotypes to the genotypes. Thus, we ensured that the trait was unrelated to the genotype data, and both the transmission of the traits and the genotypes followed the laws of Mendelian inheritance within each pedigree.

3.1.6.5 Simulation scenarios We did 500 and 1,000 replicates to estimate power and type I error rates, respectively. For each replicate, nonpolymorphic variants, if any, were removed from the subsequent analysis, for they were not informative in estimation. To simulate more practical situations in reality, we examined several distinct scenarios in which approximately 40% or 30% of the total number of variants were randomly assigned to be risk ones either with or without a mixture of 20% protective variants. To achieve and demonstrate more comparable scenarios, we designed two schemes. First, we particularly subset the 30% of the risk variants from those 40% ones, and kept the same set of 20% protective variants for the scenarios containing common and rare variants. Second, to further investigate the influence of rarity of the genetic variants on the MAF-dependent models, within the same 263 variants extracted from the pool we randomly assigned the other set of risk and protective variants by keeping the proportions identical to those in the first scheme.

While our statistics are developed mainly for rare variant analysis, we are still interesting in evaluating how the type I error rates would behavior if there is a combination of common and rare variants. Specifically, we simulated additional but similar scenarios as we did for rare variants, but allowed common variants included in the same genetic region as designated previously. Besides 263 rare variants, there were 41 common variants added in, resulting in a total number of 304 variants.

3.1.6.6 Choice of *norder* and *nbasis* There are two parameters we can specify when using the B-Spline or the Fourier basis to weight the genotypes. One parameter is *norder*, which determines the degree of basis functions. It shapes the form of the basis function by

the order of the curve (linear, quadratic, cubic, ...). The other parameter is *nbasis*, which controls the number of nodes used in the polynomial interpolation. Finding an optimal combination of *norder* and *nbasis* in the analysis is a challenging procedure with the ad hoc solutions depending on the number and rarity of genetic variants.

In our simulation study for rare variants, we considered $norder = 4$ and $nbasis = 5$ to deeply reduce the dimension for both the GFLMM and the embedded approach. For a combination of common and rare variants, we kept using $norder = 4$ and $nbasis = 5$ for the GFLMM, but recognized that type I error rates were inflated in our embedded approach if we kept using the same parameters as we did for rare variants. To enable better calibration, we chose $norder = 6$ and $nbasis = 19$ for the B-Spline basis when smoothing genotypes in the embedded approach.

3.1.6.7 Other statistics for comparison To further evaluate the performance of our proposed methods, we made a transverse comparison by exploring the other different statistics previously published. Besides the retrospective kernel and burden tests, we also included family-based SKAT (famSKAT) and F-SKAT tests in the simulation study. The famSKAT statistic is developed based on the idea of SKAT, with the capability to handle pedigree data. The extension lies in the inclusion of random effects for familial correlation [Chen et al., 2013]. Although famSKAT was initially described for the analysis of quantitative traits, we treated 0 and 1 as quantitative traits, and made famSKAT an appropriate approach for comparison. F-SKAT, on the other hand, can be considered another extension of SKAT to analyze related individuals with dichotomous traits [Yan et al., 2015]. It is based on a kernel machine regression, and within a framework of generalized linear mixed models. For all the statistics in comparison, we ran both weighted and unweighted versions. The weights were based on a function of MAF estimated by $1/\sqrt{\text{MAF} \times (1 - \text{MAF})}$ [Madsen and Browning, 2009]. Beta density weights were also used when retrospective kernel and burden statistics were computed [Wu et al., 2011].

3.1.7 Real Data Analysis

To assess the behavior of our statistics in real situations, we conducted genome-wide association analysis in the UCLA/Pitt family-based study. As discussed in Chapter 2, the same cleaned exome chip data set was used. Missing genotypes were replaced by 0, 1, or 2, denoting the copy number of minor alleles. The probabilities of the replacement with these values were determined by the non-missing genotype frequency. All those genes with at least two polymorphic variants within gene boundaries were included in the analysis. In accordance with the performance of the statistics in the simulation study, ad hoc analysis plans were made to better handle the genes with different number of variants.

3.1.7.1 Analysis of a combination of common and rare variants With the inclusion of the common variants in the data set, we first focused on the combination of common and rare variants, and explored the AMD susceptibility genes by using our statistics discussed so far in this work. We drew Q-Q plots to examine the similarity of the observed and expected values.

According to the simulation results, our GFLMM performed well under $norder = 4$ and $nbasis = 5$ for both rare variants or the combination of common and rare variants. Correspondingly, for the analysis of the real exome chip data with a combination of common and rare variants, we kept using a constant set of $norder = 4$ and $nbasis = 5$. Particularly, for the genes with more than 5 polymorphic variants, we set $norder = 4$ and $nbasis = 5$, and for the genes with 2 to 5 polymorphic variants, we set $norder = 3$ and $nbasis = 3$ to reduce the dimension.

As for the embedded approach, we picked a more complicated strategy when running the analysis. Since only the B-Spline basis was used to smooth the genotypes, there was no restriction on the parity of *nbasis*. For the genes with 2, 3 to 6, and 7 to 24 polymorphic variants, we set $norder = nbasis = 2$, $norder = nbasis = 3$, and $norder = nbasis = 6$, respectively. For the genes with more than 25 polymorphic variants, we set $norder = 6$, and let *nbasis* be the smallest integer $\geq 1/4 \times$ number of variants. Particularly, for those genes containing a large number of variants, we chosen the smallest *nbasis* value that could

maximize the dimension reduction, but still retain $\bar{G}/2 < 1$, where $\bar{G}$ was the mean of the genotypes for the variant across all of the individuals as mentioned in Section 3.1.5.

3.1.7.2 Analysis of rare variants With the exclusion of the common variants, we were then interested in exploring the rare variants. We reran the analysis by using the GFLMM and embedded approaches. For the GFLMM, we used the same choice of $norder = 4$ and $nbasis = 5$ as we did for a combination of common and rare variants in Section 3.1.7.1. For the embedded approach, per the simulation study, we considered it appropriate to accomplish weighting by simply setting $norder = 4$ and $nbasis = 5$ for those genes containing more than 4 polymorphic variants. For those small-size genes with 2 to 3 polymorphic variants, we set $norder = nbasis =$ number of variants.

3.1.8 Investigate the Behavior of Kernel-based Statistics in Real Data Analysis

In Chapter 2, we discovered and discussed the issue that kernel-based approach might generate smaller p-values than burden tests (see the Q-Q plots in Section 2.2.2 and the discussions in Section 2.3). To examine the similar issue for the GFLMM and the embedded approaches, we draw Q-Q plots similar to the ones we plotted in Section 2.2.2. For the GFLMM, we focused on the comparison between the usage of the B-Spline basis and the Fourier basis. For the embedded approach, we were interested in investigating the difference between the kernel-based and burden tests. We analyzed the scenarios of both common and rare variants and only rare variants separately.

3.2 RESULTS AND CONCLUSIONS

3.2.1 Simulation Results

Within 25 two- or third- generation families, we sampled a total of 228 related individuals. Type I error rates and power levels were estimated under both fixed and MAF-dependent OR models. For each model, we varied the proportion of risk variants by 40% or 30%, and

switched the inclusion of 20% protective variants. The empirical p-values were estimated at the nominal levels of 0.05 and 0.01, respectively.

In this section, we presented simulation results for the type I error rates and power levels by bar plots. The notations of the statistics and scenarios labeled in the plots and mentioned in the dissertation were defined in Table 8.

Table 8: Summary of the notations in the plots

Notations	Descriptions and Interpretations
flmBS	GFLMM approach with the B-Spline basis
flmFR	GFLMM approach with the Fourier basis
flmBS_kernel	embedded kernel-based approach with the B-Spline basis
flmBS_burden	embedded burden tests with the the B-Spline basis
kernel_BT	retrospective kernel-based approach with weights based on Beta distribution
burden_BT	retrospective burden tests with weights based on Beta distribution
kernel_MB	retrospective kernel-based approach with Madsen-Browning weights
burden_MB	retrospective burden tests with Madsen-Browning weights
kernel_UW	retrospective kernel-based approach with equal weights
burden_UW	retrospective burden tests with equal weights
famSKAT_W	family-based SKAT with MAF-dependent weights
famSKAT_UW	family-based SKAT with equal weights
FSKAT_W	F-SKAT with MAF-dependent weights
FSKAT_UW	F-SKAT with equal weights
2.0/0.8	Fixed OR models with OR = 2.0 for risk variants, and 0.8 for protective variants.
m10	MAF-dependent OR models with OR = $\frac{\log c}{4}\,\lvert\log_{10}\mathrm{MAF}\rvert$, where $c = 10$ Accordingly, a similar notation also applies to $c = 3$ or $c = 4$.
40%/0%	Models with 40% risk variants and no protective variants Accordingly, a similar notation also applies to 30% risk variants.
40%/20%	Models with a mixture of 40% risk variants and 20% protective variants Accordingly, a similar notation also applies to 30% risk variants.

3.2.1.1 Type I error rates We ran 1,000 replicates for each scenario when simulating the type I error rates which were presented and illustrated by Figures A.1-A.9 (see Appendix A) and Figures 9-17. The estimated p-values for the different statistics were plotted in vertical bars with various colors by different scenarios. We denoted nominal levels of 0.05 and 0.01 by solid lines. Exact 95% confidence intervals were superimposed onto the plots using dashed lines to indicate the upper and lower bounds of the estimated p-values. Although rare variants were of particular interest to this dissertation, we also examined the behavior of

our proposed statistics with a combination of rare and common variants. We also noted that the genetic model described in Section 3.1.6.3 for simulation was appropriate for estimating type I error rates with a combination of rare and common variants.

For flmBS and flmFR derived from the GFLMM, Rao's score test statistics showed conservative behaviors. Overall, the empirical type I error rates were within the 95% confidence interval of the nominal values, with a couple of exceptions. We noted that the potential inflation occurred in those scenarios with only risk variants under the MAF-dependent models where $c = 10$ or $c = 4$, regardless of rare or a combination of common and rare variants (m10-40%/0% and m4-40%/0% in Figures A.1, A.3, 9, 11, A.4, A.6, 12, and 14). These scenarios reflect the ones where a gene contains rare variants with extremely high ORs. Despite unlikely situations, to figure out the reasons, we simulated additional scenarios in which we retained the same genetic region, but re-sampled the risk and protective variants while keeping the same proportion as previously we did. For these scenarios subsequently simulated, the type I error rates showed no sign of inflation. Compared with the prior set of variants, the risk variants in the re-sampled set had lower mean MAF values, suggesting that for fixed $norder = 4$ and $nbasis = 5$, MAF of risk variants would have an impact on the performance of flmBS and flmFR.

For the embedded approaches, we chose different *norder* and *nbasis* as described in Section 3.1.7.1 when handling rare and a combination of common and rare variants. The behavior of flm_kernel and flm_burden for rare variants under $norder = 4$ and $nbasis = 5$ was conservative, with the type I error rates within the anticipated 95% confidence interval for most scenarios. With common variants mixed in, we recognized that the similar setting of $norder = 4$ and $nbasis = 5$ would result in the inflation of the type I error rates. Further examinations led us to higher order B-splines and more nodes with $norder = 6$ and $nbasis = 19$. Most of the type I error rates were within the anticipated range with one exception of m4-40%/0%, which was slightly high above the upper bound (Figures 12 and 13). There was a palpable difference in specifying *norder* and *nbasis* between rare variants and a combination of common and rare variants. When common variants mixed in, the embedded approaches called for a higher order B-Spline with more nodes for smoothing.

The other statistics for comparison, including kernel-based approach, burden tests, famSKAT, and F-SKAT, were supposed to well calibrate the type I error rates according to their original discussions, but surprisingly not F-SKAT. Based on our simulation results, the type I error rates estimated by FSKAT_W and FSKAT_UW showed significant inflation in most scenarios, regardless of with fixed or MAF-dependent ORs (as in Figures A.2 and 11), or for rare or a combination of common and rare variants (as in Figures A.2 and A.4). Retrospective techniques, for both kernel-based approach and burden tests, produced relatively stable type I error rates close to the nominal values, despite some slight underestimates for kernel-based approach in a few scenarios (as in Figures 9 and 12). While treating dichotomous traits as continuous ones, famSKAT resulted in conservative type I error rates in our simulation study, especially in those scenarios with fixed ORs and MAF-dependent ORs with a small c value (as in Figures A.3 and A.6).

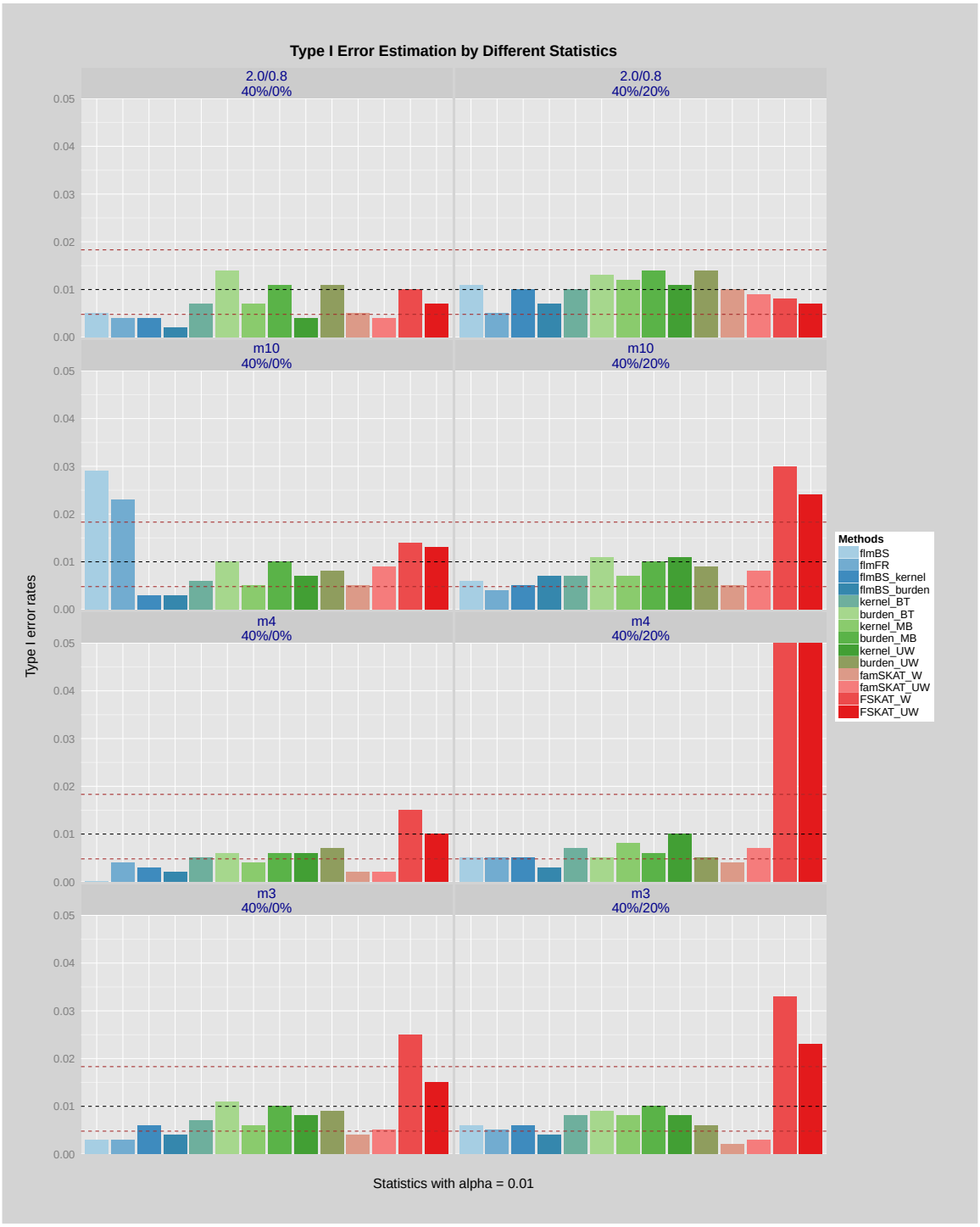

Figure 9: Type I error rates of rare risk variants ($\alpha = 0.01$, 40% risk variants). Type I error rates for 40% rare variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

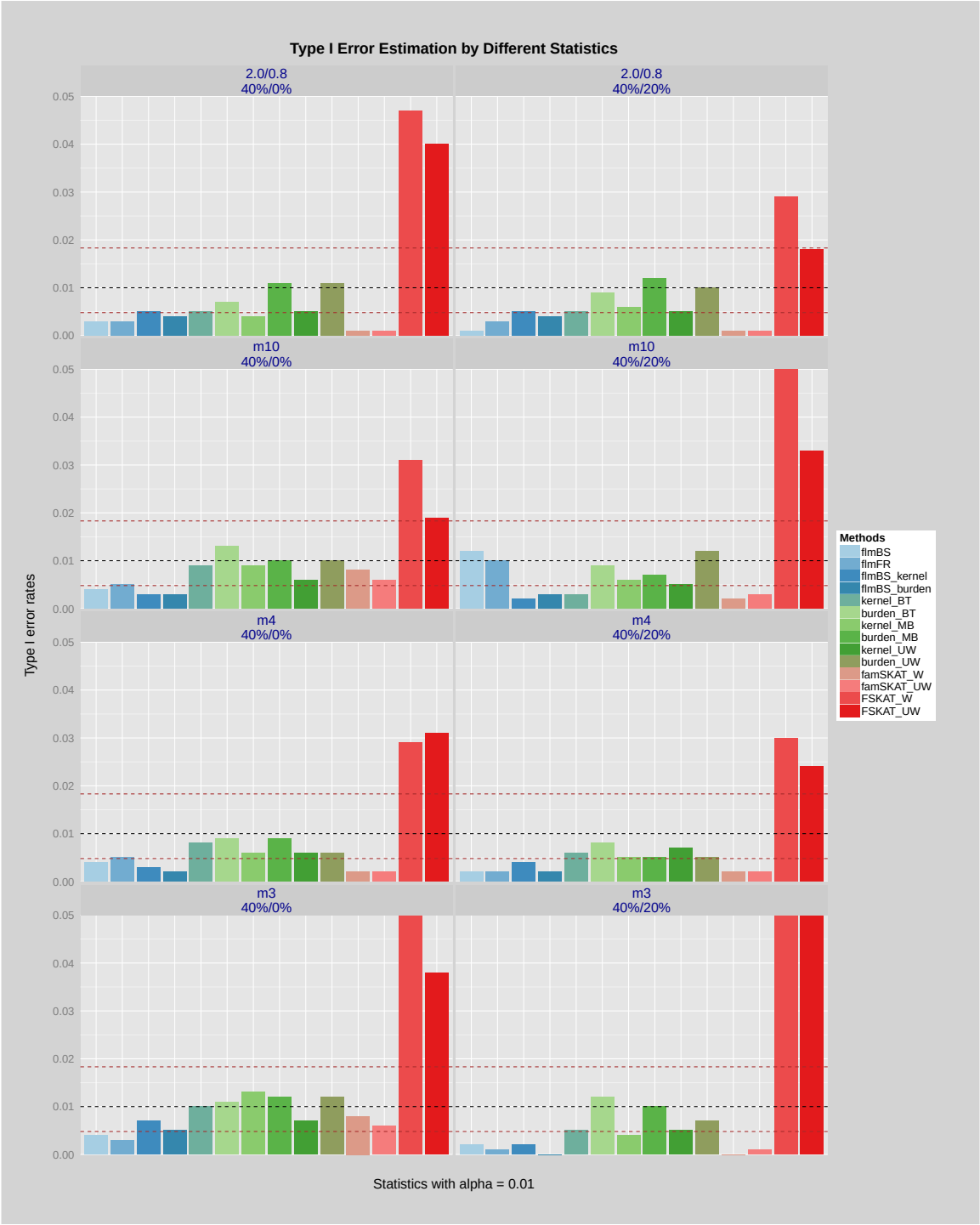

Figure 10: Type I error rates of rare risk variants ($\alpha = 0.01$, a re-sampled set of 40% risk variants). Type I error rates for a distinct set of 40% rare variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

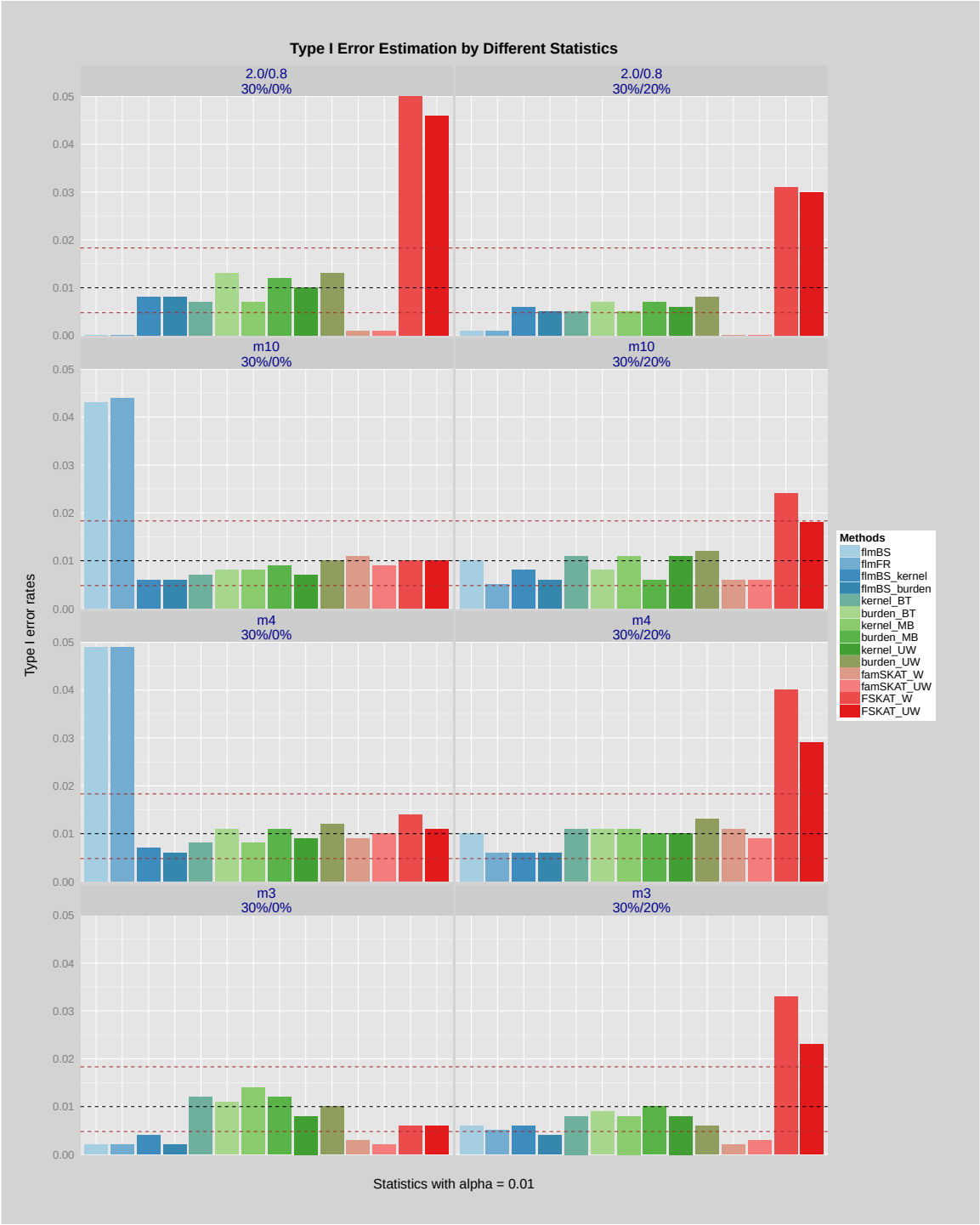

Figure 11: Type I error rates of rare risk variants (α = 0.01, 30% risk variants). Type I error rates for 30% rare variants (a subset of those in Figure 9) with or without a mixture of 20% protective variants (the same ones as in Figure 9). Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

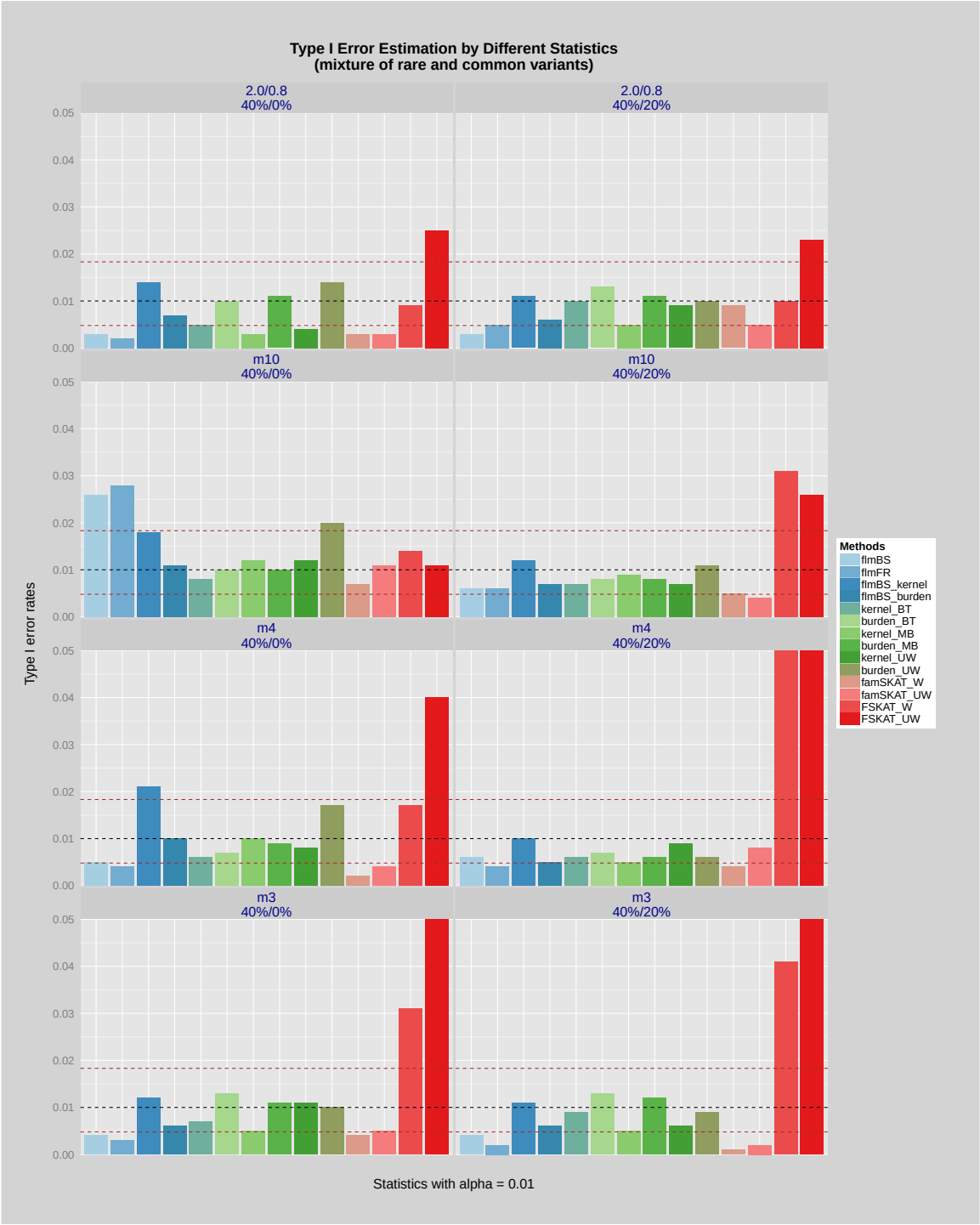

Figure 12: Type I error rates of a combination of common and rare risk variants ($\alpha = 0.01$, 40% risk variants). Type I error rates for 40% combined common and rare variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

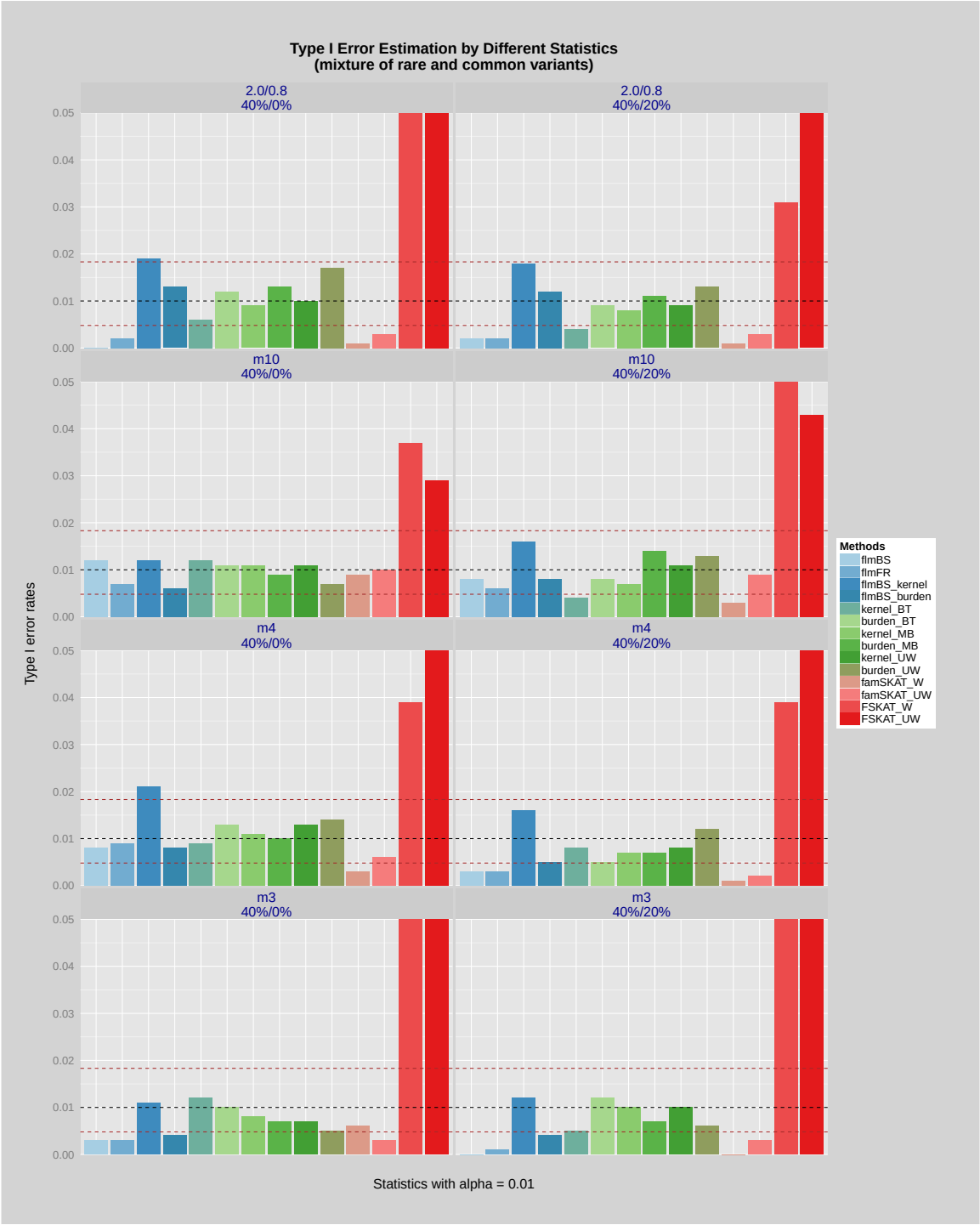

Figure 13: Type I error rates of a combination of common and rare risk variants ($\alpha = 0.01$, a re-sampled set of 40% risk variants). Type I error rates for a re-sampled set of 40% combined common and rare variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

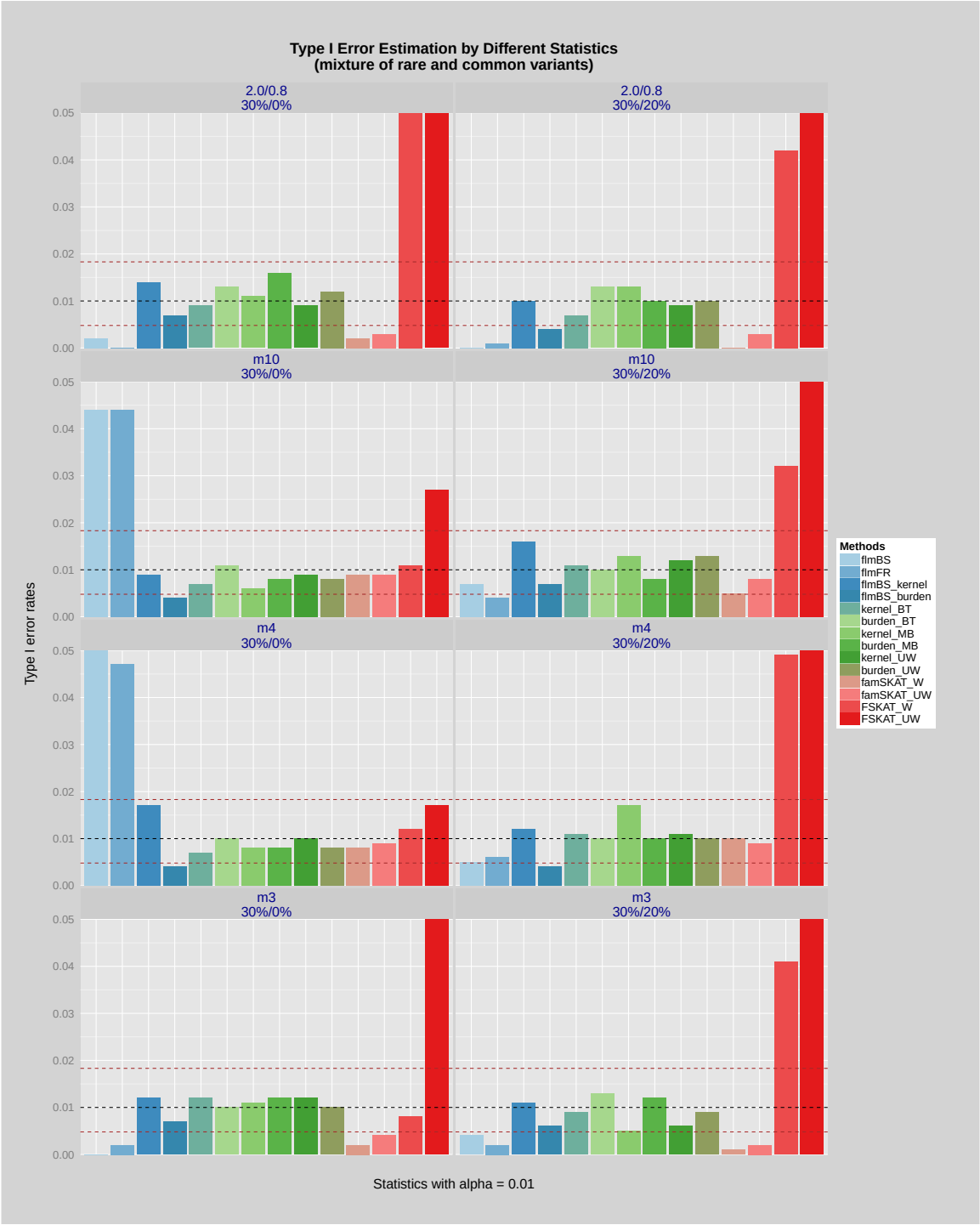

Figure 14: Type I error rates of a combination of common and rare risk variants ($\alpha = 0.01$, 30% risk variants). Type I error rates for 30% rare variants (a subset of those in Figure 12) with or without a mixture of 20% protective variants (the same ones as in Figure 12). Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

3.2.1.2 Power levels We ran 500 replicates for each scenario when estimating empirical power levels which were presented and illustrated by Figures A.7-17. For comparison purposes, the standard error of each statistic at a 95% confidence level was superimposed onto the bar plots to define the upper and lower bounds of the estimated power levels. We labeled "I" or "D" at the bottom of each bar to indicate the inflation or deflation of its corresponding type I error rate, so that the calibration of the type I error rates could be reflected on the presentation and interpretation of the power levels. Power levels estimations were only done for rare variants.

Since F-SKAT's type I error rates were way off the nominal values in quite a few scenarios, the measure of its power levels actually was not comparable to those of well-calibrated statistics. In general, both our family-based GFLMMs (flmBS and flmFR) and embedded approaches (flmBS_kernel and flmBS_burden), regardless of weighting by the B-Spline or the Fourier basis, outperformed famSKAT, despite the overlaps of the 95% confidence levels in the scenarios with lower c values and the presence of protective variants, indicating a commensurate behavior of these statistics in m4- or m3- models with only rare variants.

Compared with the retrospective kernel-based approach or burden tests, neither flmBS nor flmFR exhibited an overall competitive behavior. Although our family-based GFLMMs might produce satisfying empirical power levels in the m10- or m4- models with only rare variants (Figures A.9, 16, and 17), evidence was lacking as to a more promising approach than the retrospective kernel-based approach or burden tests in most scenarios.

For the embedded approach, we were aware of the dissimilarity in power levels between the kernel-based approach and burden tests. After embedding the FLM-smoothed genotypes into the retrospective kernel-based approach, we might be able to increase power levels especially for the scenarios with fixed ORs (Figures A.8-17) or those including only risk variants with MAF-dependent ORs (as in Figures A.8 and 16). For burden tests, however, although they were competitive in fixed OR models or MAF-dependent models with only risk variants, weighting and smoothing genotypes failed to bring any advantages, or were even detrimental to power levels (as in Figures A.7 and A.8).

In summary, we estimated empirical power levels for rare variants with our proposed GFLMMs and the embedded approach, and compared them with the retrospective kernel-

based and burden test, famSKAT, and F-SKAT. Both our proposed method and embedded approach had an overall advantage over either famSKAT or F-SKAT, regardless of the genetic models we assumed. In contrast, neither flmBS nor flmFR outperformed the kernel-based or burden statistics. Besides, we considered it worthwhile to embed smoothed genotypes in the kernel-based approach, but not the burden tests, in view of the performance of flmBS_kernel than the corresponding kernel_BT, kernel_MB, or kernel_UW. By contrast, although the measure of flmBS_burden had nearly the same power levels as flmBS_kernel, we did not have sufficient evidence to argue that it was superior to the corresponding burden_BT, burden_MB, or burden_UW.

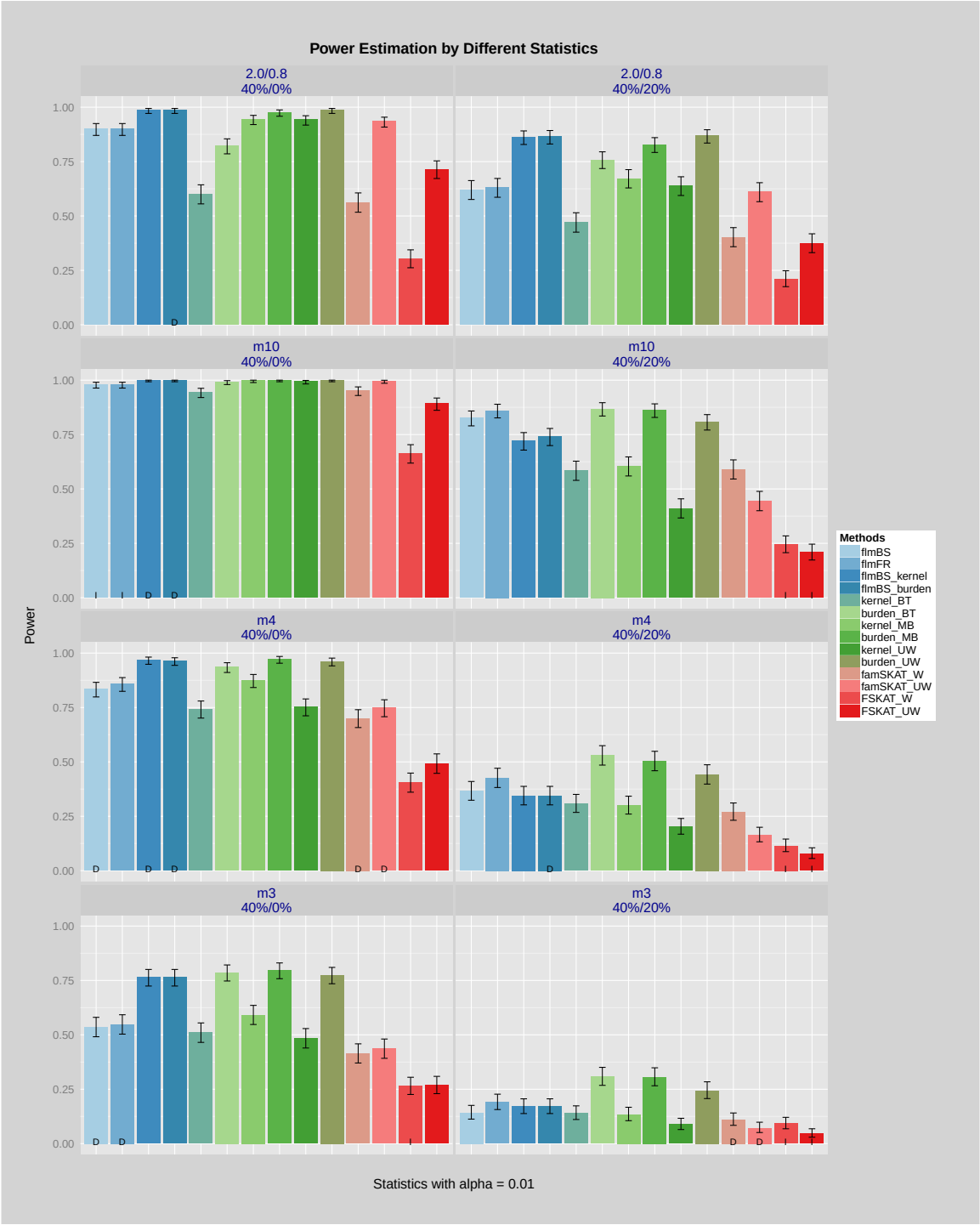

Figure 15: Power levels of rare risk variants ($\alpha = 0.01$, 40% risk variants). Power levels for 40% rare risk variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. "I" or "D" at the bottom of each bar indicates the inflation or deflation of its corresponding type I error rate. Notations of statistics and scenarios are defined in Table 8.

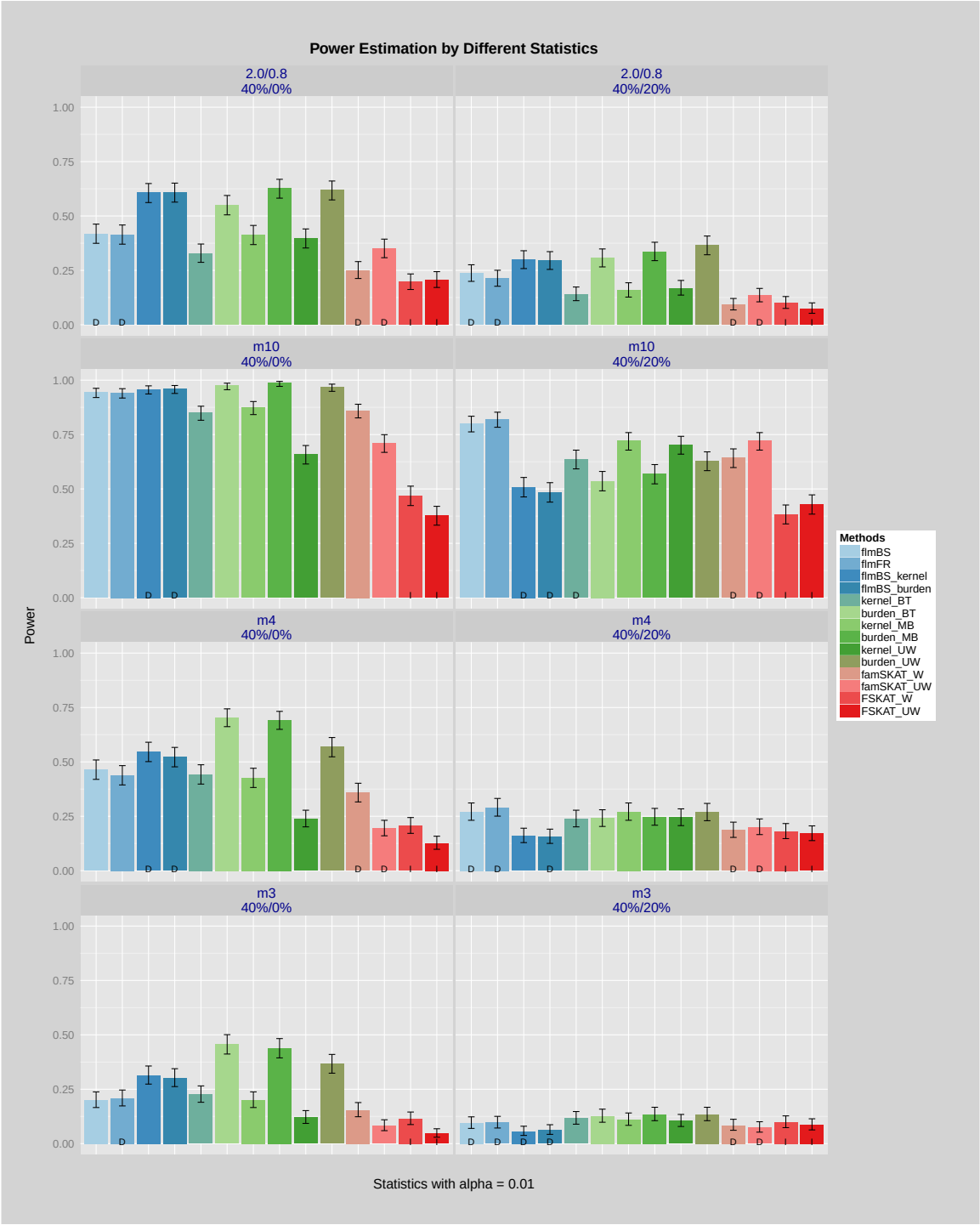

Figure 16: Power levels of rare risk variants ($\alpha = 0.01$, a distinct set of 40% risk variants). Power levels for a re-sampled set of 40% rare risk variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. "I" or "D" at the bottom of each bar indicates the inflation or deflation of its corresponding type I error rate. Notations of statistics and scenarios are defined in Table 8.

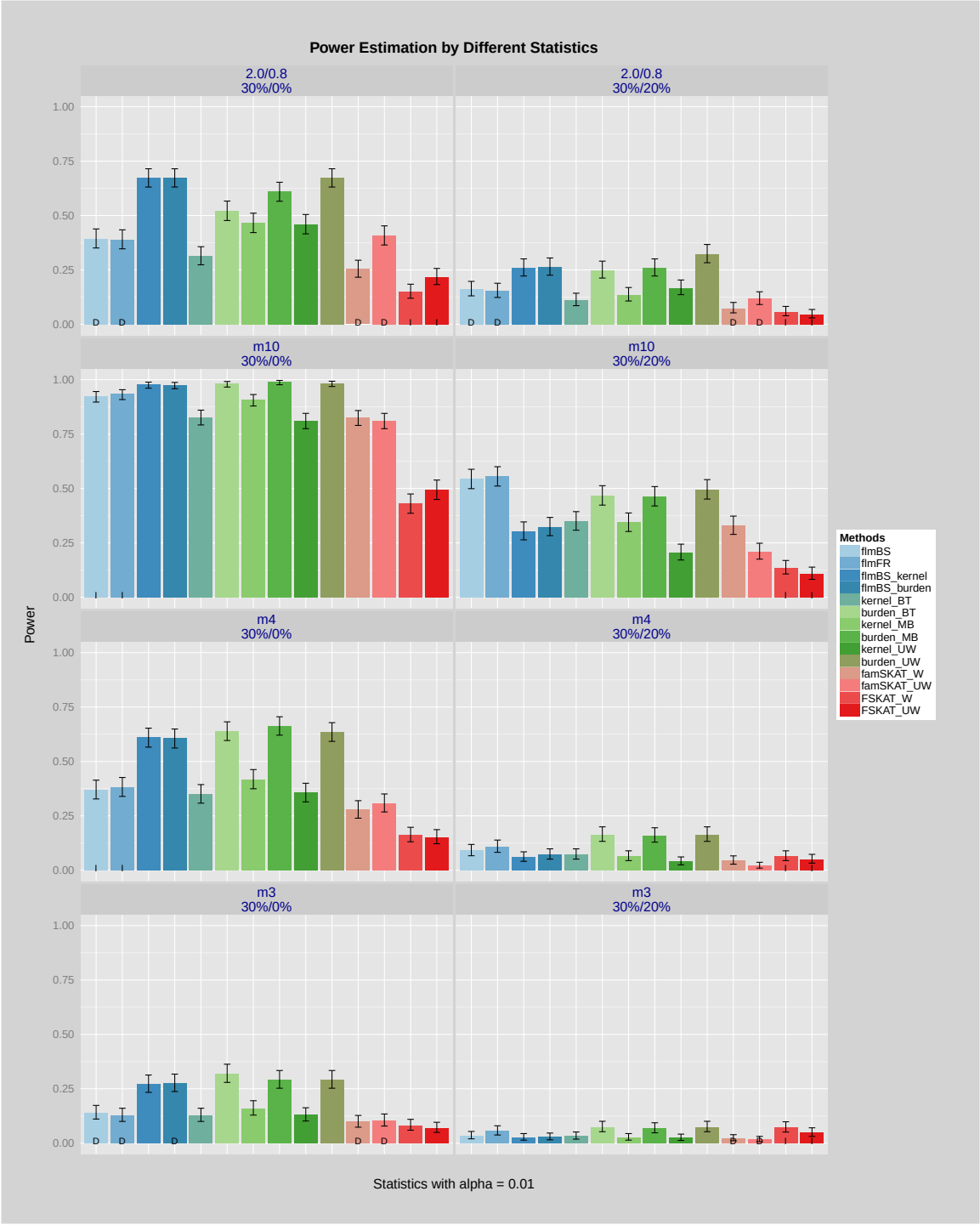

Figure 17: Power levels of rare risk variants ($\alpha = 0.01$, 30% risk variants). Power levels for 30% rare risk variants (a subset of those in Figure 15) with or without a mixture of 20% protective variants (the same ones as in Figure 15). Nominal type I error rate is 0.01. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. "I" or "D" at the bottom of each bar indicates the inflation or deflation of its corresponding type I error rate. Notations of statistics and scenarios are defined in Table 8.

3.2.2 Real Exome Chip Data Analyses on AMD

According to the simulation results, we applied our proposed GFLMM and the embedded approach to the analysis of a combination of common and rare variants with an ad hoc choice of *norder* and *nbasis* as discussed in Sections 3.1.7.1 and 3.1.7.2. We identified 16,844 genes containing at least two polymorphic variants. The results with at least one test statistic associated with a significant p-value were presented in Table 9. The corresponding p-values of the ordinary retrospective kernel and burden tests were also listed for a comparison purpose. We confirmed the strong association between AMD and *CFH* and *ARMS2*. Besides *CFH* and *ARMS2*, the other genes were not previously reported as AMD risk genes. Interestingly, *CARD9* reached genome-wide significance only through solitary flmBS_kernel ($p = 5.416 \times 10^{-10}$). This gene was previously shown to be involved in a pathway related to the mechanisms of sterile inflammation which played a pivotal role in the etiology of AMD [Strasser et al., 2012, Montezuma et al., 2007]. Another interesting significant association signal was related to *ADAMTS6* by flmFR ($p = 2.784 \times 10^{-6}$), a gene once reported with the expression in the retinal pigment epithelial cells, possibly indicating an influence on the etiology of AMD [Bevitt et al., 2003].

For the analysis of rare variants, we identified 14,849 genes containing at least two polymorphic variants. The results with at least one test statistic associated with a significant p-value were presented in Table 10. Unsurprisingly, *CFH* and *ARMS2* did not show up in the table, which was consistent with the results discussed in Section 2.2.2. We observed that *ACO1* popped out only in the rare variant analysis associated with significant flmFR ($p = 3.449 \times 10^{-17}$, see Table 10). *ACO1*, also known as *IRP1*, was previously associated with AMD through iron metabolism [Synowiec et al., 2012], suggesting supportive evidence of the association between AMD and *ACO1*.

However, the results in Table 10 should be interpreted with caution, since all of the significant p-values were returned only by flmFR (see the "flmFR" column in Table 10). In addition, we also noticed that most of these genes included a relatively small number of variants ranging from 4 to 7, with the exception of *CEP250* with 10 variants. This might lead to a suspicion that flmFR, FLM-smoothed by the Fourier basis, may not be a reliable

method for the analysis of those genes that include only a few variants. Actually, smoothing genotype data might not be necessary for those genes containing the number of variants ≤ 7, or even ≤ 10, since the GFLMM, like the generalized FLM, assumes that genotype data within a certain genetic region on a chromosome are a realization of a stochastic process. Furthermore, if a large genetic region contains only a few variants, modeling them as a functional curve is not a reasonable analysis approach.

In our exome chip data set, we noted that one of the *CFH* transcripts contained seven rare variants but failed to be identified in the rare variant analysis by any statistics. To further unveil these rare variants, we examined their association on an individual level by conducting the M_{QLS} test [Thornton and McPeek, 2007]. According to the summaries in Table 11, we observed that none of these rare variants reached genome-wide significance, leading to a consistent conclusion drawn from the gene-based association analysis with different approaches and statistics.

Table 9: GFLMM and embedded approaches for a combination of common and rare variants

chr	gene	nvariant	length (kb)	start (bp)	end (bp)	flmBS	flmFR	flmBS_kernel	flmBS_burden	kernel_MB	burden_MB
chr1	***CFH***	3	50	196621007	196670695	$\mathbf{2.364 \times 10^{-07}}$	$\mathbf{2.313 \times 10^{-07}}$	$\mathbf{5.145 \times 10^{-09}}$	$\mathbf{8.490 \times 10^{-09}}$	$\mathbf{1.537 \times 10^{-08}}$	$\mathbf{1.477 \times 10^{-07}}$
chr1	***CFH***	15	96	196621007	196716634	$\mathbf{1.638 \times 10^{-08}}$	$\mathbf{2.219 \times 10^{-08}}$	$\mathbf{1.320 \times 10^{-12}}$	$\mathbf{3.266 \times 10^{-11}}$	$\mathbf{8.253 \times 10^{-12}}$	$\mathbf{6.527 \times 10^{-10}}$
chr1	*GBP3*	4	16	89472359	89488549	2.060×10^{-03}	9.657×10^{-05}	$\mathbf{1.250 \times 10^{-10}}$	4.599×10^{-06}	7.883×10^{-03}	2.707×10^{-01}
chr1	*MOB3C*	5	7	47073386	47080805	5.016×10^{-01}	4.876×10^{-01}	$\mathbf{5.930 \times 10^{-11}}$	$\mathbf{1.250 \times 10^{-06}}$	4.405×10^{-01}	2.743×10^{-01}
chr1	*MOB3C*	5	9	47073386	47082563	5.016×10^{-01}	4.876×10^{-01}	$\mathbf{5.930 \times 10^{-11}}$	$\mathbf{1.250 \times 10^{-06}}$	4.405×10^{-01}	2.743×10^{-01}
chr2	*CCDC142*	7	10	74699958	74710357	1.844×10^{-01}	$\mathbf{1.143 \times 10^{-06}}$	5.810×10^{-01}	7.524×10^{-01}	3.415×10^{-01}	9.636×10^{-01}
chr5	*ADAMTS6*	9	333	64444562	64777704	1.952×10^{-01}	$\mathbf{2.784 \times 10^{-06}}$	4.449×10^{-01}	6.774×10^{-01}	1.142×10^{-01}	5.227×10^{-01}
chr8	*GSDMD*	6	5	144640476	144645231	8.442×10^{-02}	$\mathbf{8.471 \times 10^{-13}}$	1.356×10^{-01}	5.790×10^{-01}	1.143×10^{-01}	6.278×10^{-01}
chr8	*GSDMD*	6	10	144635556	144645231	8.442×10^{-02}	$\mathbf{8.471 \times 10^{-13}}$	1.356×10^{-01}	5.790×10^{-01}	1.143×10^{-01}	6.278×10^{-01}
chr8	*PSD3*	5	282	18384812	18666405	3.248×10^{-04}	$\mathbf{7.246 \times 10^{-07}}$	8.830×10^{-03}	1.456×10^{-02}	2.894×10^{-04}	5.794×10^{-03}
chr8	*WDYHV1*	4	25	124428964	124454260	5.076×10^{-01}	$\mathbf{1.281 \times 10^{-09}}$	7.769×10^{-01}	7.586×10^{-01}	4.982×10^{-01}	8.667×10^{-01}
chr9	*CARD9*	5	10	139258407	139268133	2.504×10^{-02}	1.607×10^{-02}	$\mathbf{5.416 \times 10^{-10}}$	3.587×10^{-05}	1.215×10^{-02}	2.674×10^{-01}
chr10	***ARMS2***	2	3	124214178	124216868	$\mathbf{7.274 \times 10^{-14}}$	$\mathbf{2.920 \times 10^{-12}}$	$\mathbf{3.047 \times 10^{-15}}$	$\mathbf{1.396 \times 10^{-14}}$	$\mathbf{6.536 \times 10^{-13}}$	$\mathbf{1.118 \times 10^{-10}}$
chr11	*OR9Q1*	14	158	57791352	57949038	1.532×10^{-01}	1.575×10^{-01}	$\mathbf{1.013 \times 10^{-10}}$	7.047×10^{-05}	8.593×10^{-02}	2.733×10^{-01}
chr10	*LIPN*	7	17	90521162	90537999	4.346×10^{-01}	$\mathbf{2.345 \times 10^{-06}}$	6.830×10^{-01}	7.280×10^{-01}	3.653×10^{-01}	5.071×10^{-01}
chr10	*PTPRE*	4	38	129845812	129884164	9.188×10^{-01}	$\mathbf{3.887 \times 10^{-07}}$	8.629×10^{-01}	6.279×10^{-01}	5.146×10^{-01}	5.820×10^{-01}
chr11	*CTTN*	6	38	70244611	70282690	1.103×10^{-02}	$\mathbf{1.556 \times 10^{-21}}$	1.511×10^{-01}	9.454×10^{-02}	5.488×10^{-03}	2.188×10^{-03}
chr14	*DNAAF2*	5	10	50091891	50101948	1.043×10^{-01}	$\mathbf{1.604 \times 10^{-06}}$	7.573×10^{-02}	5.009×10^{-02}	6.711×10^{-02}	3.041×10^{-02}
chr14	*PTGR2*	4	34	74318620	74352168	2.114×10^{-01}	3.338×10^{-01}	$\mathbf{5.701 \times 10^{-15}}$	$\mathbf{1.220 \times 10^{-11}}$	1.338×10^{-01}	7.326×10^{-02}
chr14	*PTGR2*	4	34	74318533	74352168	2.114×10^{-01}	3.338×10^{-01}	$\mathbf{5.701 \times 10^{-15}}$	$\mathbf{1.220 \times 10^{-11}}$	1.338×10^{-01}	7.326×10^{-02}
chr14	*PTGR2*	4	34	74318546	74352168	2.114×10^{-01}	3.338×10^{-01}	$\mathbf{5.701 \times 10^{-15}}$	$\mathbf{1.220 \times 10^{-11}}$	1.338×10^{-01}	7.326×10^{-02}
chr17	*UBE2Z*	3	21	46985730	47006422	3.892×10^{-02}	4.073×10^{-02}	$\mathbf{1.331 \times 10^{-07}}$	$\mathbf{5.365 \times 10^{-07}}$	5.234×10^{-03}	4.513×10^{-03}
chr18	*ZNF521*	5	290	22641887	22932214	3.508×10^{-02}	$\mathbf{6.043 \times 10^{-07}}$	3.857×10^{-02}	1.839×10^{-02}	1.842×10^{-03}	2.971×10^{-03}

GFLMM and embedded approaches for a combination of common and rare variants. Genes were reported with at least one test statistic associated with a p-value $< 2.97 \times 10^{-6}$ (highlighted in boldface). Known AMD risk genes were highlighted in boldface.

Table 10: GFLMM and embedded approaches for rare variants

chr	gene	nvariant	length (kb)	start (bp)	end (bp)	flmBS	flmFR	flmBS_kernel	flmBS_burden	kernel_MB	burden_MB
chr2	*CCDC142*	7	10	74699958	74710357	1.844×10^{-01}	$\mathbf{1.143 \times 10^{-06}}$	5.156×10^{-01}	7.451×10^{-01}	3.415×10^{-01}	9.636×10^{-01}
chr7	*ADAM22*	4	269	87563565	87832204	1.745×10^{-01}	$\mathbf{9.168 \times 10^{-07}}$	5.709×10^{-01}	9.601×10^{-01}	1.798×10^{-01}	1.250×10^{-01}
chr7	*ADAM22*	4	248	87563565	87811428	1.745×10^{-01}	$\mathbf{9.168 \times 10^{-07}}$	5.709×10^{-01}	9.601×10^{-01}	1.798×10^{-01}	1.250×10^{-01}
chr8	*GSDMD*	6	5	144640476	144645231	8.442×10^{-02}	$\mathbf{8.471 \times 10^{-13}}$	1.386×10^{-01}	6.219×10^{-01}	1.143×10^{-01}	6.278×10^{-01}
chr8	*GSDMD*	6	10	144635556	144645231	8.442×10^{-02}	$\mathbf{8.471 \times 10^{-13}}$	1.386×10^{-01}	6.219×10^{-01}	1.143×10^{-01}	6.278×10^{-01}
chr9	***ACO1***	6	66	32384600	32450832	6.670×10^{-01}	$\mathbf{3.449 \times 10^{-17}}$	6.867×10^{-01}	5.639×10^{-01}	5.199×10^{-01}	4.760×10^{-01}
chr10	*LIPN*	7	17	90521162	90537999	4.346×10^{-01}	$\mathbf{2.345 \times 10^{-06}}$	6.573×10^{-01}	7.158×10^{-01}	3.653×10^{-01}	5.071×10^{-01}
chr11	*CTTN*	6	38	70244611	70282690	1.103×10^{-02}	$\mathbf{1.556 \times 10^{-21}}$	2.762×10^{-01}	1.229×10^{-01}	5.488×10^{-03}	2.188×10^{-03}
chr11	*SAA2-SAA4*	5	17	18252901	18270221	3.977×10^{-01}	$\mathbf{3.651 \times 10^{-19}}$	7.622×10^{-01}	7.760×10^{-01}	3.644×10^{-01}	4.063×10^{-01}
chr12	*METTL25*	4	121	82752275	82873016	3.200×10^{-01}	$\mathbf{2.097 \times 10^{-07}}$	2.726×10^{-01}	3.157×10^{-01}	2.084×10^{-01}	5.348×10^{-02}
chr20	*CEP250*	10	57	34043222	34099803	1.984×10^{-01}	$\mathbf{9.127 \times 10^{-07}}$	1.836×10^{-01}	2.650×10^{-01}	4.106×10^{-01}	5.183×10^{-01}

GFLMM and embedded approaches for rare variants. Genes were reported with at least one test statistic associated with a p-value $< 3.37 \times 10^{-6}$ (highlighted in boldface). Known AMD risk genes were highlighted in boldface.

Table 11: Single-variant association analysis for the rare risk variants within *CFH*

variant	gene	chr	position (bp)	major/minor	MAF	M_{QLS}
rs141336681	*CFH*	chr1	196642221	[A/C]	0.0013	0.035
rs143237092	*CFH*	chr1	196695675	[A/C]	0.0006	0.960
rs515299	*CFH*	chr1	196706677	[A/C]	0.0026	0.330
rs149474608	*CFH*	chr1	196709816	[T/G]	0.0058	0.160
rs145975787	*CFH*	chr1	196709833	[T/C]	0.0006	0.770
rs534399	*CFH*	chr1	196711067	[A/C]	0.0026	0.330
rs35274867	*CFH*	chr1	196712596	[A/T]	0.0097	0.710

M_{QLS} tests for the rare risk variants within *CFH* identified in Table 10. Interestingly, none of these rare variants reached a significant p-value $< 2.09 \times 10^{-6}$ for the single-variant analysis by using M_{QLS}.

Similar to what we did in Chapter 3, we assessed the behavior of the statistics for the real data analysis by drawing the Q-Q plots (Figures 18-21), separated by rarity of the variants within the genes in our analysis. The Q-Q plots were also drawn with those AMD risk genes removed for comparison. We dropped 10% of the most significant results when estimating the inflation factor. For the GFLMM approach, regardless of variant rarity, the B-Spline basis led to a little higher λ_{GC} value than the Fourier basis, but we did not observe a huge behavior discrepancy (Figures 18 and 20). For the embedded approach, no inflation in the λ_{GC} value was observed for the analysis of both common and rare variants (Figure 19) or for the analysis of only rare variants (Figure 21). Compared with the ordinary retrospective kernel test in Chapter 2 (see Figures 7 and 8), our results suggested that the embedded approach with the FLM-smoothed genotypes might lower λ_{GC} values by reducing the dimension of genotype data in both kernel-based approach and burden tests. After we filtered out the AMD risk genes, the behavior of the Q-Q plots was similar in terms of the λ_{GC} values.

Particularly, for the analysis of only rare variants by using flmBS and flmFR, we noticed an obvious difference between applying the B-Spline basis and the Fourier basis (see Figure 20). For flmBS, almost all of the points were within the 95% confidence interval, while for flmFR, quite a few points were highly above the reference line, even the risk AMD genes were excluded. This might give supportive evidence to indicate that smoothing by the Fourier basis might not work well, consistent with the results summarized in Table 10.

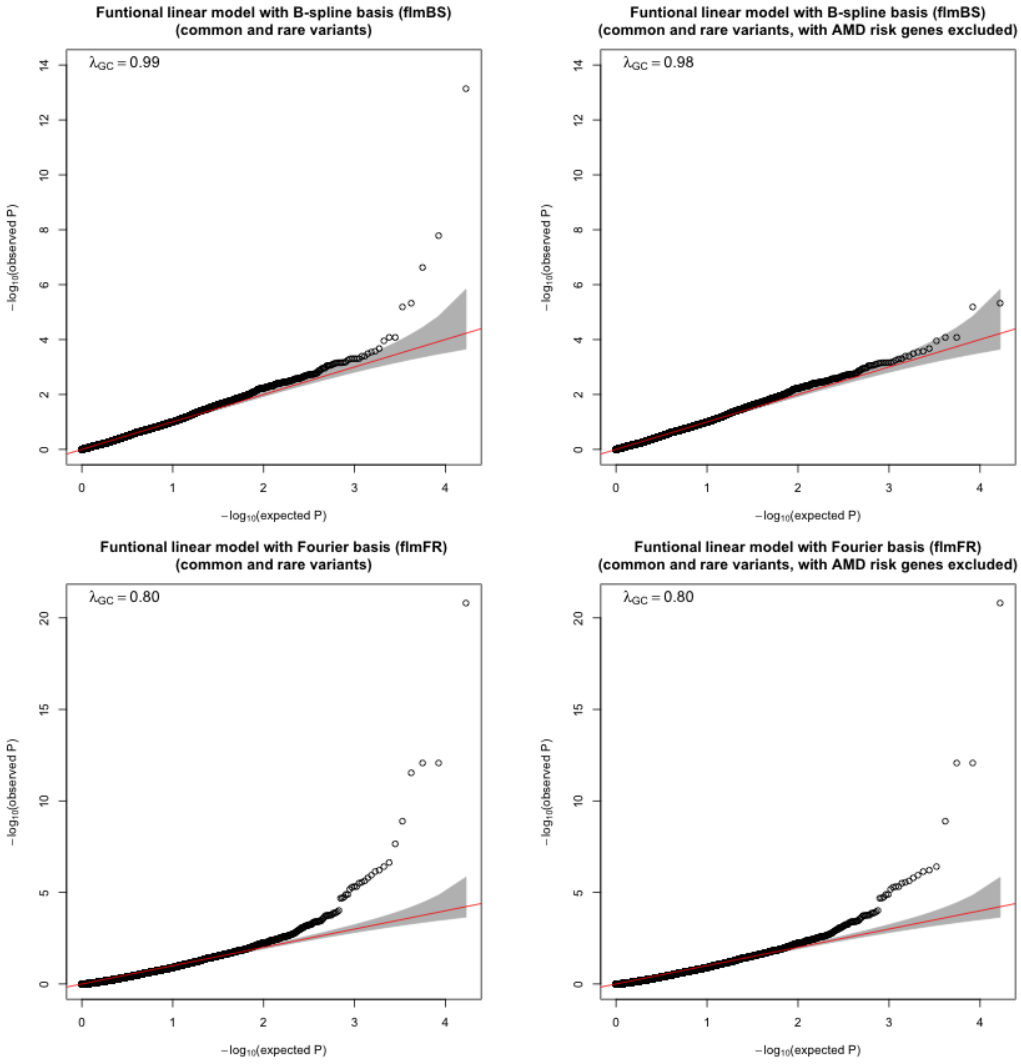

Figure 18: Q-Q plots of the GFLMM with the B-Spline (flmBS) and the Fourier basis (flmFR) for a combination of common and rare variants. Q-Q plots for the GFLMM with the B-Spline (flmBS) and the Fourier basis (flmFR) were illustrated on the top and bottom row, respectively. Left and right column showed the genes with and without AMD risk genes, respectively. The genes contained a combination of common and rare variants. The λ_{GC} value was estimated and superimposed onto the plots to denote genomic inflation factor, indicating how far the points were away from the reference line.

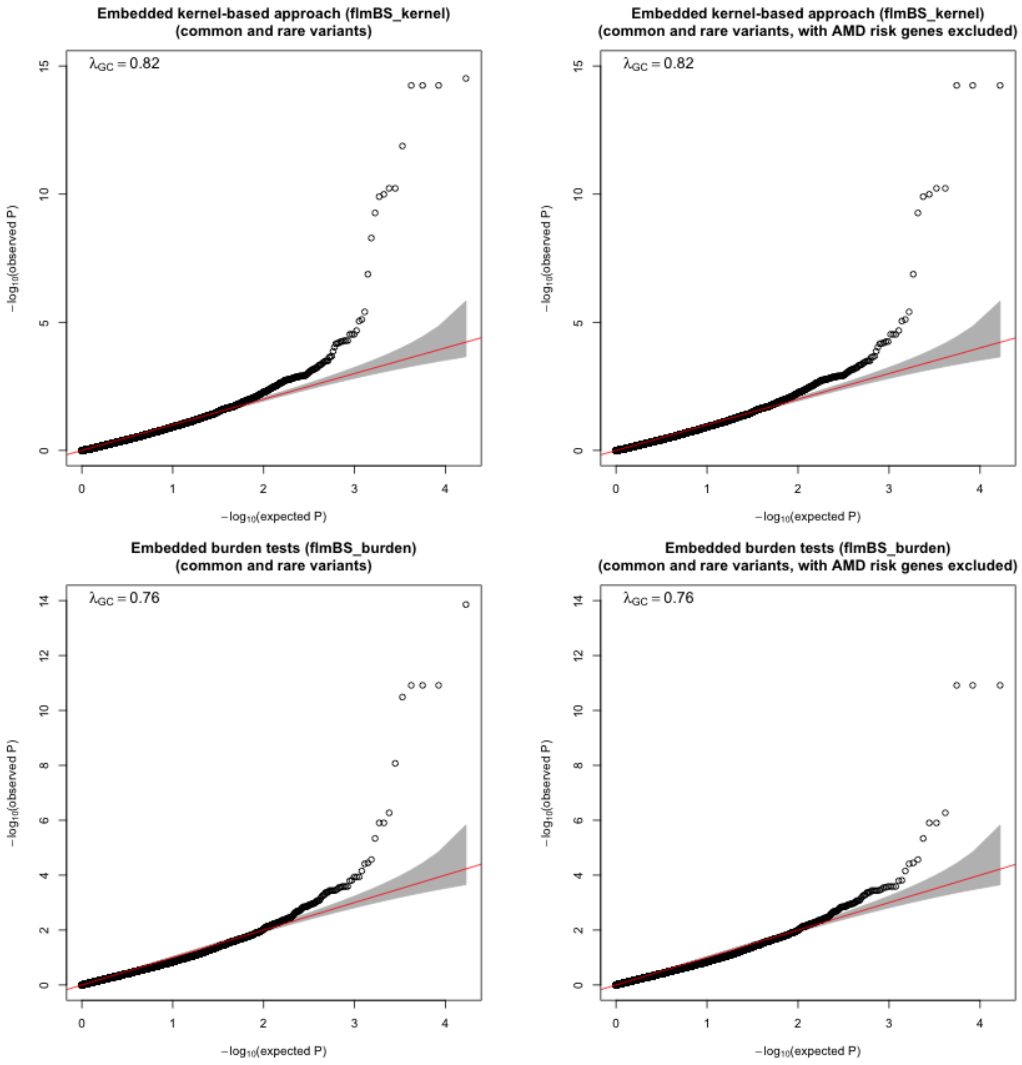

Figure 19: Q-Q plots of the embedded kernel-based approach (flmBS_kernel) and burden tests (flmBS_burden) with a combination of common and rare variants. Q-Q plots for the embedded kernel-based approach (flmBS_kernel) and burden tests (flmBS_burden) were illustrated on the top and bottom row, respectively. Left and right column showed the genes with and without AMD risk genes, respectively. The genes contained a combination of common and rare variants. λ_{GC} value was estimated and superimposed onto the plots to denote genomic inflation factor, indicating how far the points were away from the reference line.

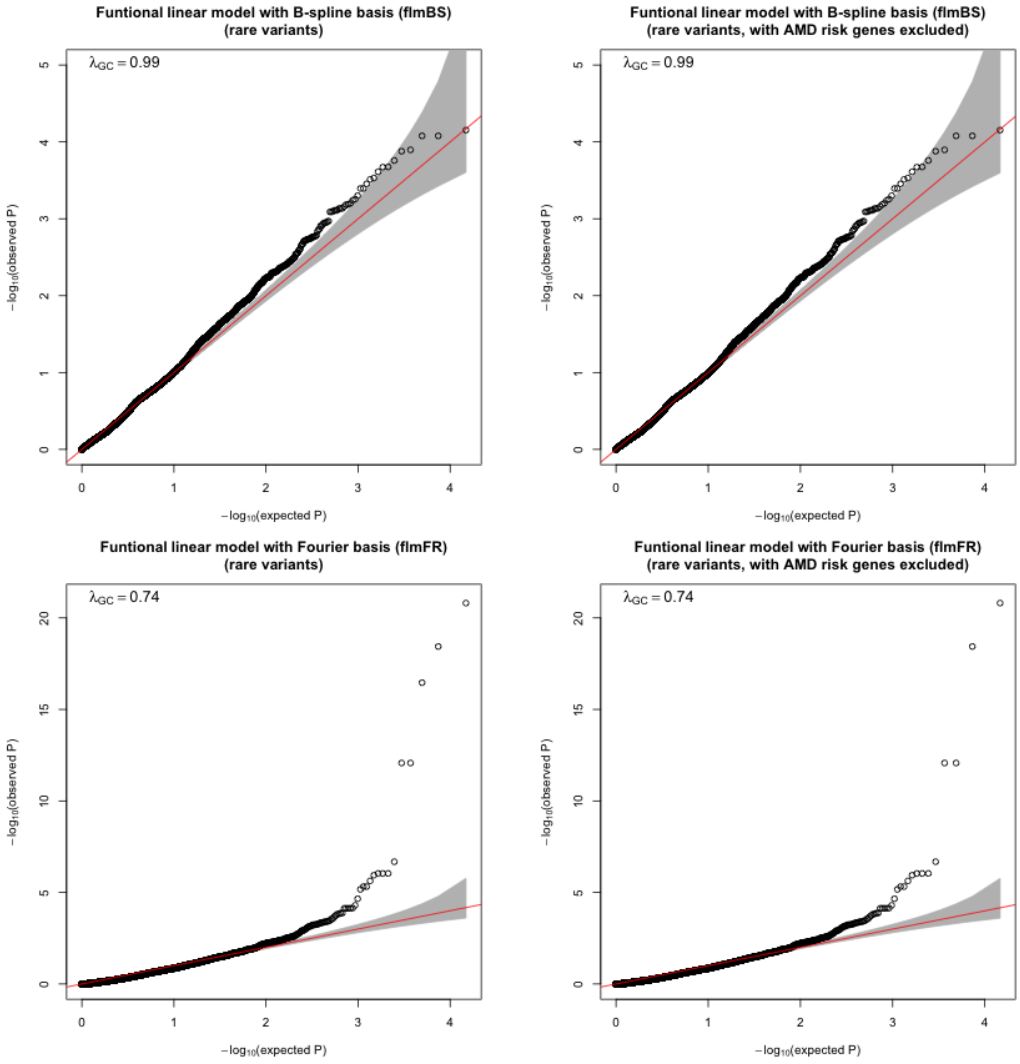

Figure 20: Q-Q plots of the GFLMM with the B-Spline (flmBS) and the Fourier basis (flmFR) for only rare variants. Q-Q plots for the GFLMM with the B-Spline (flmBS) and the Fourier basis (flmFR) were illustrated on the top and bottom row, respectively. Left and right column showed the genes with and without AMD risk genes, respectively. The genes contained only rare variants. The λ_{GC} value was estimated and superimposed onto the plots to denote genomic inflation factor, indicating how far the points were away from the reference line.

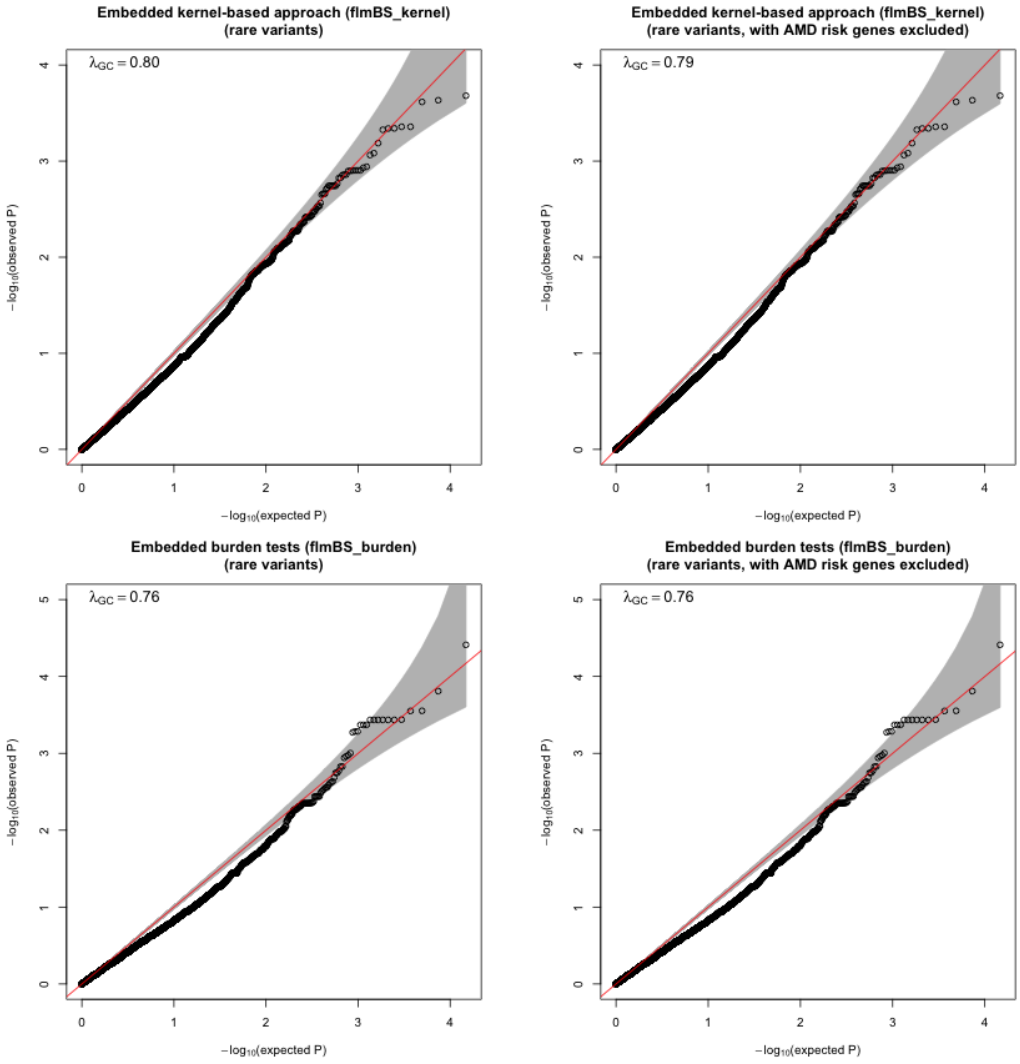

Figure 21: Q-Q plots of the embedded kernel-based approach (flmBS_kernel) and burden tests (flmBS_burden) with only rare variants. Q-Q plots for the embedded kernel-based approach (flmBS_kernel) and burden tests (flmBS_burden) were illustrated on the top and bottom row, respectively. Left and right column showed the genes with and without AMD risk genes, respectively. The genes contained only rare variants. λ_{GC} value was estimated and superimposed onto the plots to denote genomic inflation factor, indicating how far the points were away from the reference line.

3.3 DISCUSSION

In this chapter, we have developed the GFLMM to evaluate the association between dichotomous traits and disease susceptibility genes in related subjects. It is an important extension of the generalized FLM to the analysis of pedigree data, with the capability of analyzing both common and rare variants. To estimate the parameters of our proposed model and compute gene-based test statistics, we have integrated out the random polygenic effect term, and constructed Rao's score test statistics in the presence of nuisance parameters. The parameter estimation procedures are an extension of the methods discussed by Stanhope and Abney [2012]. Their approach is limited to analyzing single variants, while our GFLMM is able to take multiple variants into account, and construct gene-based tests.

Previously, Stanhope and Abney developed the GLOGS program to estimate the parameters of generalized linear mixed models [Stanhope and Abney, 2012]. Based on our analyses and testing, we saw several flaws in the original program, and fixed them. First, we added a step-halving process into the Gauss-Newton algorithm. Sometimes the algorithm converges in a very slow manner, even our initial input value is close to the true one. Since we desire to maximize the likelihood under the parameter estimation procedure, we do not anticipate that the likelihood estimated by the current iteration is less than the one estimated in the preceding iteration. To make the program more efficient, we set up an *if* condition to control whether or not to invoke the step-halving method. Second, we consider the stopping criterion in the original program controversial, because it was determined by the absolute difference of first derivatives of the parameters estimated from two adjacent iterations. To clearly describe the problem, we simplified the discussion to one dimension, and gave a counterexample in Section 3.1.4.2. In general, the first derivative describes the local monotonicity of a function at a certain point, but may not be an appropriate stopping criterion for function optimization (maximization of the likelihood in our case). Third, the original GLOGS program uses Bayesian updates to approximate the integral. However, this results in a single weight in the example data provided by Stanhope and Abney. We argue that it may be problematic to use a single weight to approximate a multi-dimensional integral of a complicated function. We instead use equal weights that no longer depend on the iterations

to improve the original GLOGS program, resulting in the cancellation of the weights in both numerators and denominators in Equation 3.7. By using two small sample data sets, we have tested our modifications, compared, and verified the integration results with two existing R packages. We have demonstrated in Table 7 that our modifications to the original GLOGS program are necessary and very important to approximate a multi-dimensional integral.

In terms of the parameter estimation, we observed that our modified GLOGS program tended to return a lower estimate of σ than 0.2, a constant we set for the polygenic effect in the simulation study. Those estimates in several simulation scenarios can be as small as 10^{-4} or 10^{-5}, indicating a trivial polygenic effect as compared with the genetic effect. This may be caused by two reasons. On one hand, there may be still an issue in the original GLOGS program when estimating the polygenic effect, and we failed to identify it in our study. On the other hand, while simulating pedigrees, we only allowed those pedigrees with at least 1 pair of affected siblings for the evaluation of type I error rates and power levels. This ascertainment criterion may make the genetic effect dominant, and thus marginalize the polygenic effect. It may be interesting to simulate pedigrees with a relatively large polygenic effect (but not as robust as the genetic effect) in future work, and track if our modified GLOGS program can more precisely estimate the polygenic parameter.

In addition to developing the GFLMM, we have also embedded the FLM-smoothed genotypes in the retrospective kernel and burden tests. The advantage in doing this is that we do not have to be concerned about the ascertainment issue, for it is within the retrospective regression framework, and meanwhile the genotypes in the model are FLM-smoothed. It should be noted that we do not apply any weightings like we did for kernel_MB or burden_MB, since the genotypes in the model have already been weighted by the B-Spline basis.

When it comes to the choice of *norder* and *nbasis*, several factors may have an impact on it. First, the choice may be sensitive to different statistical methods. According to the simulation study with presence of both common and rare variants, the GFLMM can well calibrate type I error rates with the choice of $norder = 4$ and $nbasis = 5$ for most scenarios, while we have to specify a higher $norder = 6$ and $nbasis = 19$ for the embedded approach to analyze the same data sets simulated by an identical genetic model. Second, the choice may

be sensitive to the rarity of variants. For the embedded approach, with the common variants added in, a dramatic change in *norder* (from 4 to 6) and *nbasis* (from 5 to 19) is required to well calibrate type I error rates. Third, the choice may be sensitive to the number of variants within genes. This may not be exhibited in the simulation study due to the reason that we have fixed a genetic region of interest when simulating genotype data. However, it is highlighted in the real data analysis with genes of all kinds of length, thus containing a different number of variants. From the viewpoint of dimension reduction, it does not make much sense to keep *nbasis* as a constant regardless of varying gene sizes. Thus, we have come up with an ad hoc rule for choosing *nbasis* in the real data analysis.

We have simulated data to compare the GFLMM, the embedded approach, the ordinary retrospective kernel and burden tests, famSKAT, and F-SKAT. Commonalities of these statistical methodologies include modeling dichotomous traits, analyzing rare variants, and accounting for related people with known pedigree structures. The advantages of our suggested GFLMM are characterized by fitting discrete genotypes with a continuous curve by using a collection of smooth basis functions, and computing the exact likelihood of the model by conducting a multidimensional integration, which is different from approximating by the hierarchical likelihood or a quadratic form with the iterative weighted least squares (IWLS) algorithm. According to the simulation results, the GFLMM outperforms famSKAT and F-SKAT, but is not better than the retrospective kernel and burden statistics in terms of power levels. Inspired by the retrospective regression that can conquer the issue of modeling ascertainment [Schaid et al., 2013], we have embedded the FLM-smoothed genotypes in Schaid's approach. For a given variant, its genotype data are coded by the copy number of minor alleles with an element from the set of $\{0, 1, 2, \text{NA}\}$, which means that after smoothing the genotypes, we cannot reduce the dimension to some extent such that an average number > 2 across all of the samples.

For the real data analysis, we have confirmed the association between AMD and *CFH* and *ARMS2* by using the GFLMM, regardless of applying the B-Spline or the Fourier basis. With those genes reaching genome-wide significance (flmBS and flmFR in Tables 9 and 10), not all of them were also seen to be significant for the retrospective kernel and burden tests (kernel_MB and burden_MB in Tables 9 and 10). We have noted a few inconsistencies where

the flmFR, with FLM-smoothed by the Fourier basis, returned significant p-values associated with the genes not identified by any other statistics in Table 10. These results may provoke some suspicions regarding smoothing by the Fourier basis, although the association of *ACO1* has been discussed by some other literature. For the B-Spline smoother, this potential issue is not observed. All these observations lead us to recommend not using the Fourier basis to smooth genotype data when analyzing genes containing a small number of variants. If there are only a few variants located in a long genetic region, it may not be appropriate to assume these variants are a realization of a stochastic process and thus modeling them by a functional curve may not be a good idea.

Since the approximation of the multidimensional integral calls for intensive computations, it is time-consuming to build GFLMMs and estimate relevant the p-values. In our real data analysis, the computations were conducted on a high performance computing cluster. The cluster included 32 processors, each with a speed of 1,400 MHz. A total number of 64 genes were allowed to be analyzed simultaneously. The memory limitation was 2.6 GB for each computing core, and it took 18-20 minutes on a single processor to complete estimating the p-value for each gene, which is not desirable for a genome-wide study. The program we wrote in R to compute the score test statistics includes the part to read in the Sobol points to approximate the integral and estimate the first and second derivatives (see B.5.1). It takes time to read in the large vector containing tens of thousands of Sobol points used to approximate the multidimensional integral. The subsequent *for* loop executes the computation of the derivatives repeatedly given certain Sobol points. Since the number of Sobol points is huge, the estimation progress is slow. To speed up the process, we may consider to rewrite the code in C or consider to achieve the summation using matrix manipulation.

The statistics derived from the embedded approach are rapid to compute, which is an advantage when analyzing genome-wide data. The embedded approach is based on Schaid's retrospective regression framework, and utilizes FLM-smoothed genotype data, which is a combination of retrospective and functional data analysis. However, we should proceed with caution. We have used the embedded approach to discover the association signal of *CFH*, *ARMS2*, and *CARD9*, which we are able to find some previous publications to support. However, besides these promising findings, the association of the other genes have never

been reported before. Given their relatively small number of variants, it reminds us to have a second thought about the smoothing and reducing the dimension of the small size genes containing only a few variants. As introduced early in Chapter 1, the idea behind the GFLMM comes from treating genotyping data as a realization of a stochastic process that varies along the chromosome. If a relative long genetic region contains very few variants, this realization may be questionable, which means that we may lack enough data points to fit a reasonable functional curve.

In Chapter 2, we have discussed a potential issue related to the smaller p-values in the kernel-based approach than in the burden tests. In this chapter, we have investigated the behavior of the statistics derived from the GFLMM and the embedded approach by drawing Q-Q plots. It is not beyond our expectation that the GFLMM or the embedded burden tests would not give rise to the obvious discrepancy of the p-values between the B-Spline basis and the Fourier basis, or the kernel-based and the burden statistics. Compared with the λ_{GC} values estimated by the retrospective kernel-based approach and burden tests, the embedded methods have produced lower estimates, indicating that the embedment can attenuate the inflation related to the kernel-based test via reducing the dimension of genotype data, as long as we can find an optimal pair of *norder* and *nbasis*.

4.0 FUTURE WORK

In this chapter, we talk about future studies we consider worthwhile to follow this dissertation.

Through the discussions in Chapter 2, we discovered an issue related to the retrospective kernel regression that kernel-based approach resulted in higher λ_{GC} values than burden tests. By using our embedded approaches, the issue was attenuated because we managed to reduce the dimension of the genotype data from the number of variants to the number of basis functions we have specified. In our simulation studies (see 3.1.6), we chosen a null gene with a length of 14 kb, a median gene size. To better demonstrate the issue and our solutions, we plan to simulate more scenarios by varying gene sizes to include a different number of variants. The purpose for conducting this is to further investigate how GFLMMs perform in the genes containing different number of variants calling for a dimension reduction, especially by using the Fourier basis. For large-size genes, a reasonable choice may be 27 kb, a reported mean size of the genes from human genome [Lander et al., 2001]. For small-size genes, we can halve the length of 14 kb to include less variants in analysis. The simulation steps would be the same as we discussed in 3.1.6. We would mainly focus on the kernel-based approach and evaluate statistics and associated p-values for the ordinary retrospective regression approaches, the embedded approach we have suggested, and famSKAT as well.

Although we have presented the GFLMM allowing for covariates other than genetic effect, it is tricky and time-consuming to estimate parameters by using the GLOGS program in presence of other covariates to be adjusted in the model. The original GLOGS program limits itself to ≤ 4 covariates, and invokes different functions when estimating the models containing different number of covariates. To modify more functions in the GLOGS program to well handle the models allowing for covariates is part of our future work.

As previously discussed in Section 3.1.6.6, our GFLMM relies on two parameters, *norder* and *nbasis*. On one hand, we intend to reduce the dimension of genotype data controlled by *nbasis*. On the other hand, we have to choose a certain pair of *norder* and *nbasis* to well calibrate type I error rates while reaching high power levels. Per the results for our simulation studies, the choice of *norder* and *nbasis* may be different between the analysis of common and rare variants and the analysis of only rare variants. To seek an optimal pair of *norder* and *nbasis*, we need to simulate more scenarios with different pairs of *norder* and *nbasis*. Based on our previous discussions and Fan's early publication [Fan et al., 2014], we can set $4 \leq norder \leq 6$ and $8 \leq nbasis \leq 21$. For real data analysis, the choice may be more complicated. We can develop a more sophisticated ad hoc plan according to the size or the number of variants in all of the genes in human genome. Ideally, given an *norder*, *nbasis* may be a function of the number of variants with a certain gene. Alternatively, we can also use a genotype data set related to all AMD risk genes as previously discussed and verified, and analyze by our GFLMM and embedded approach with different sets of *norder* and *nbasis* to see whether or not the results are significant as expected.

We recognize that the number of iterations we simulated for evaluating type I error rates was 1,000, not large enough to investigate the behavior of the statistics under extremely small nominal significance levels. For the GFLMM, the computation procedures are intensive and time-consuming, preventing us from running like 100,000 iterations. On the contrary, the computation can be rapid for the embedded approach, famSKAT, and F-SKAT, which makes it possible for us to repeat more iterations to simulate type I error rates under nominal $\alpha = 0.005$ or $\alpha = 0.0001$.

Although our study is mainly focused on rare variants, to make our arguments more persuasive and demonstrative, we may simulate some scenarios with common variants only. The genetic model (see Equation 3.22) we considered to assign the dichotomous trait in the simulation study is appropriate for rare variants or a combination of common and rare variants. To simulate the scenarios with only common variants, we have to use a different strategy to assign a disease trait.

To push forward our discussions even further, we can derive the statistic from likelihood ratio test (LRT). In our study, we intended to compute the exact marginal likelihood by in-

tegrating out the random term, and then construct the score test statistics in the GFLMM. The theoretical computation is precise, while to get around the integral takes time and effort. The logic behind LRT in our case is to approximate the likelihood of the GFLMM by using some numerical algorithms, and maximize the likelihood of the model with genetic effect and the one under the null hypothesis. The parameter estimation and likelihood optimization can be efficiently completed by several existing R packages or functions like "lme4" (https://cran.r-project.org/web/packages/lme4/index.html) and the "glmmPQL" function in "MASS" (https://stat.ethz.ch/R-manual/R-devel/library/MASS/html/glmmPQL.html).

APPENDIX A

ADDITIONAL SIMULATION PLOTS

The following plots illustrate the type I error rates and the empirical power levels simulated under a nominal $\alpha = 0.05$. In general, the patterns of these plots are similar to those depicted in Section 3.2.1.

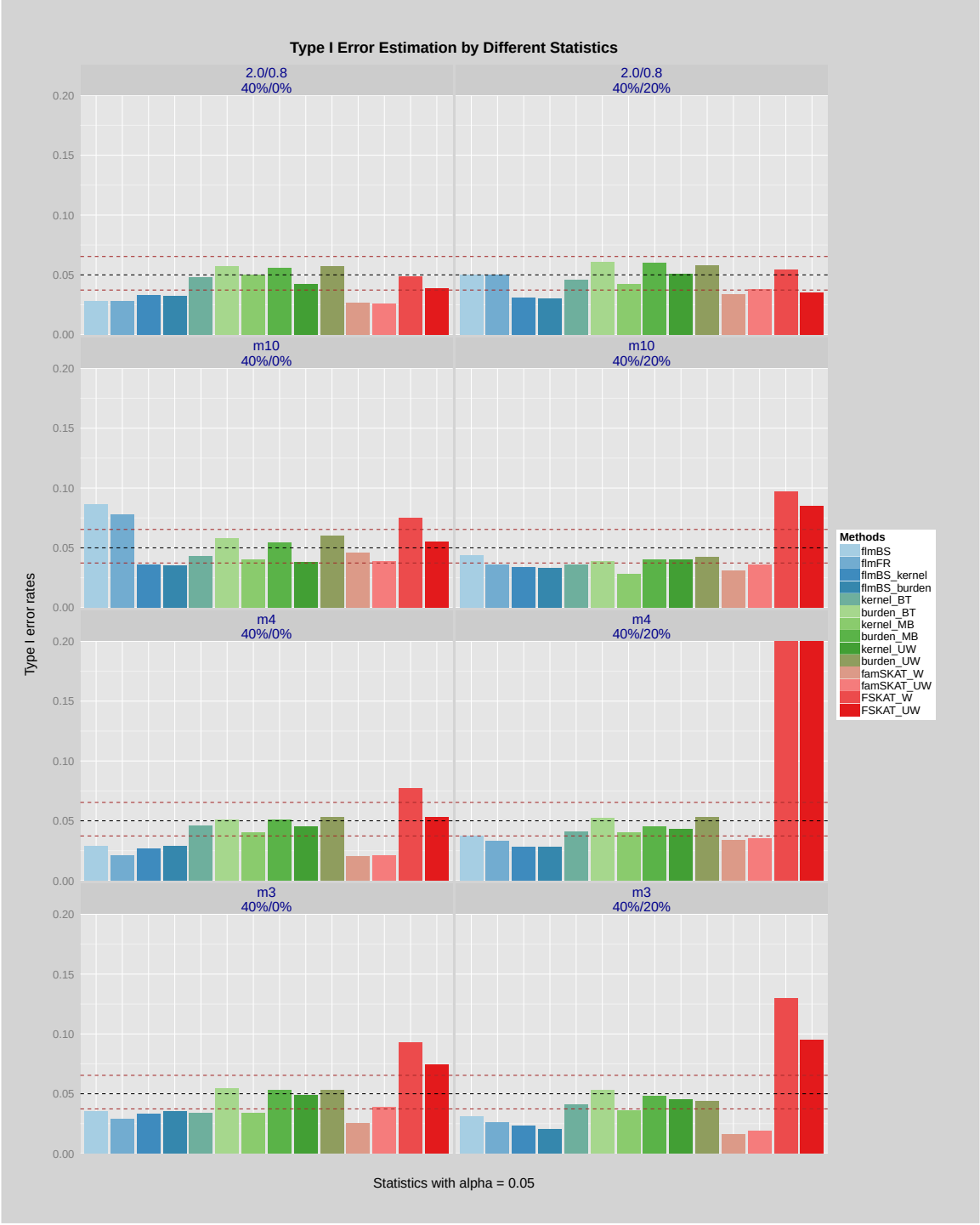

Figure A.1: Type I error rates of rare risk variants ($\alpha = 0.05$, 40% risk variants). Type I error rates for 40% rare risk variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

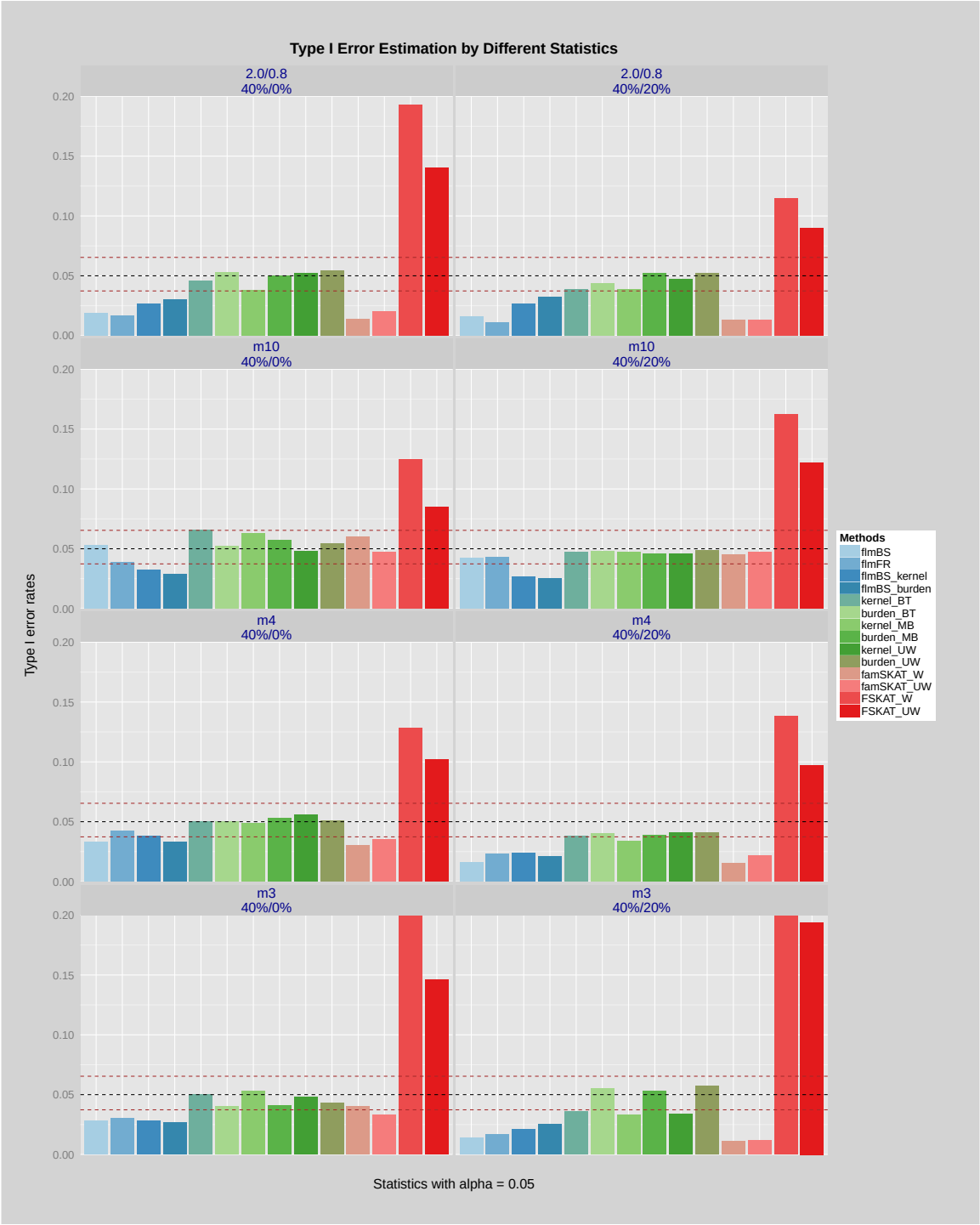

Figure A.2: Type I error rates of rare risk variants ($\alpha = 0.05$, a re-sampled set of 40% risk variants). Type I error rates for a re-sampled set of 40% rare variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

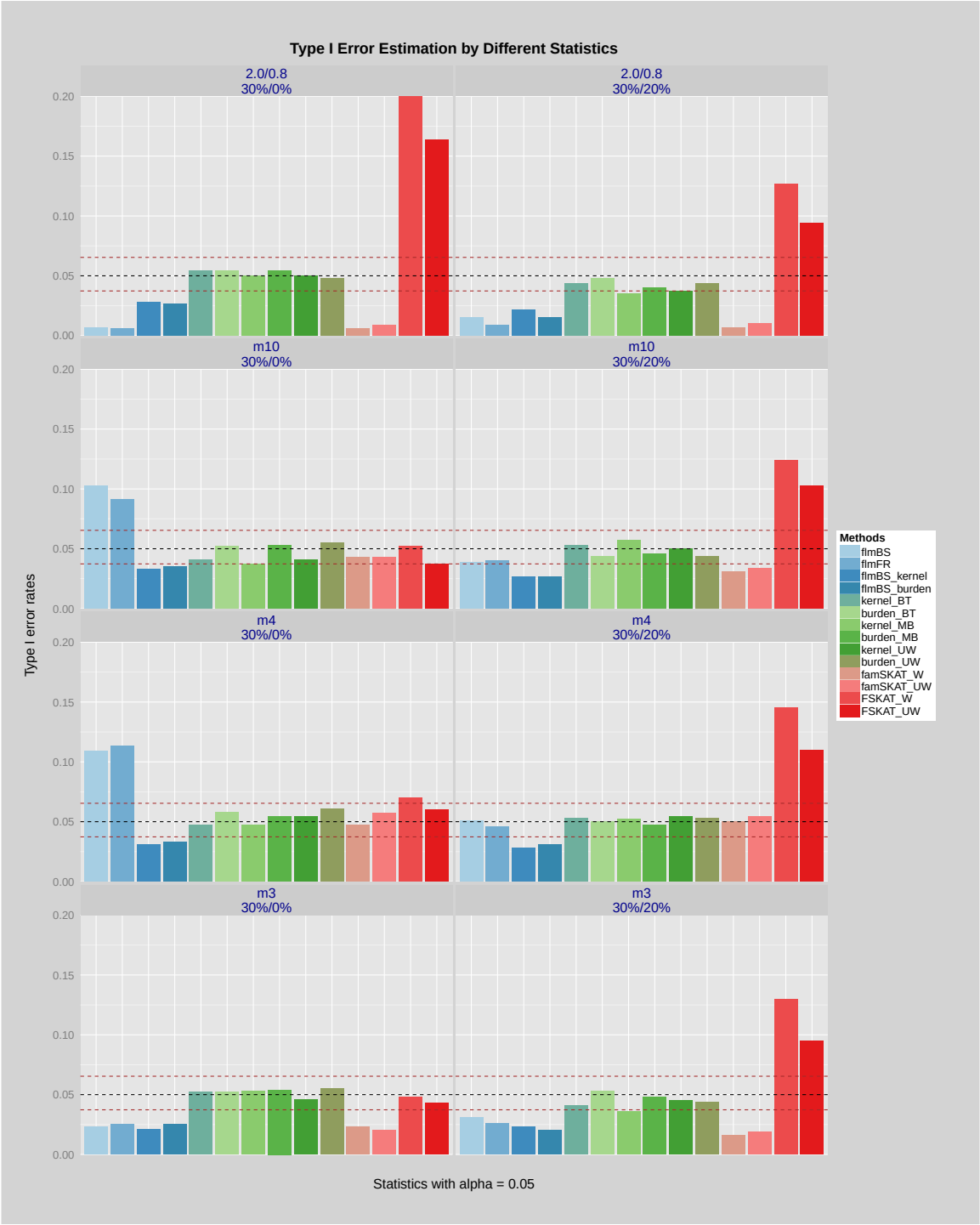

Figure A.3: Type I error rates of rare risk variants ($\alpha = 0.05$, 30% risk variants). Type I error rates for 30% rare risk variants (a subset of those in Figure A.1) with or without a mixture of 20% protective variants (the same ones as in Figure A.1). Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

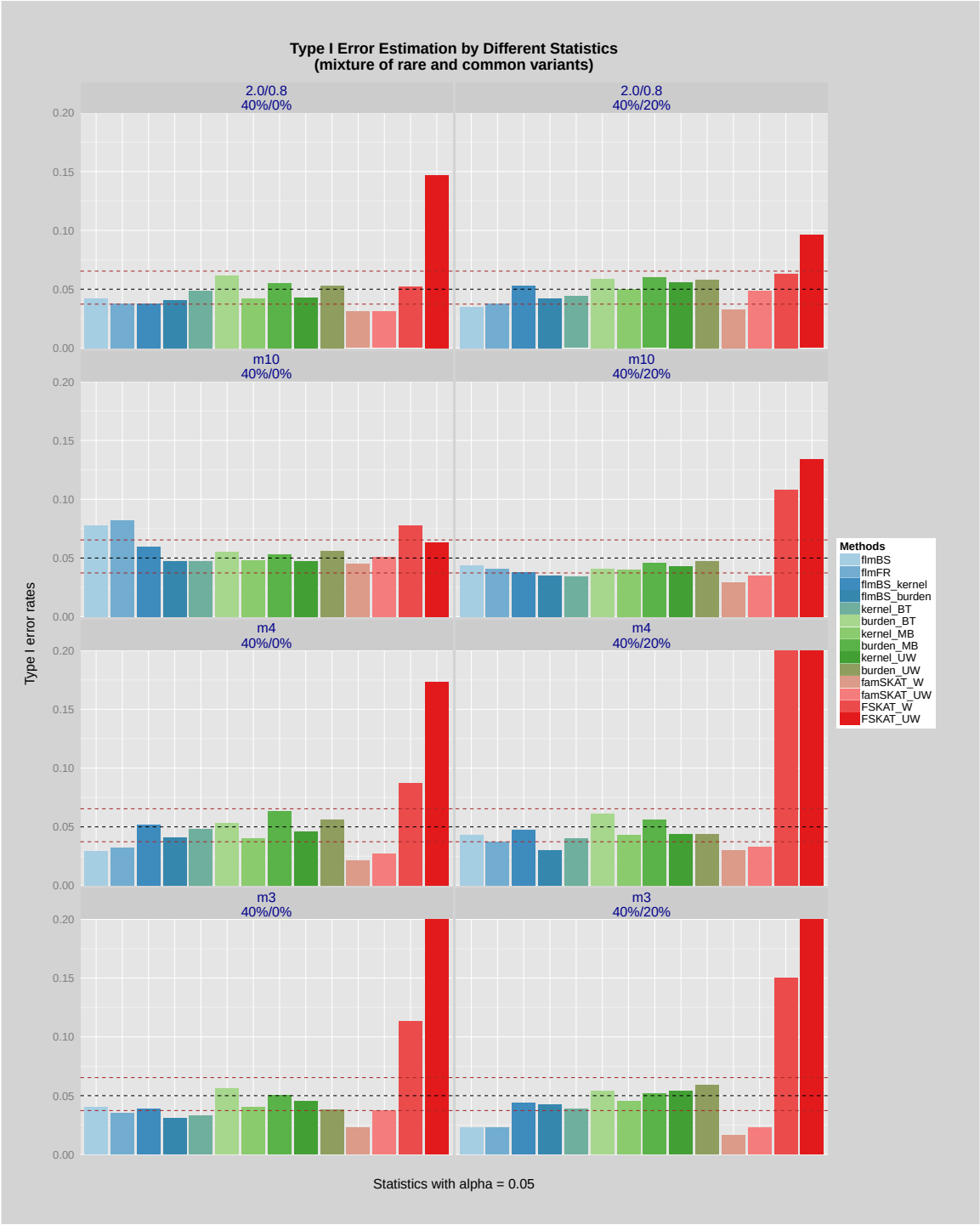

Figure A.4: Type I error rates of a combination of common and rare risk variants ($\alpha = 0.05$, 40% risk variants). Type I error rates for 40% combined common and rare risk variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

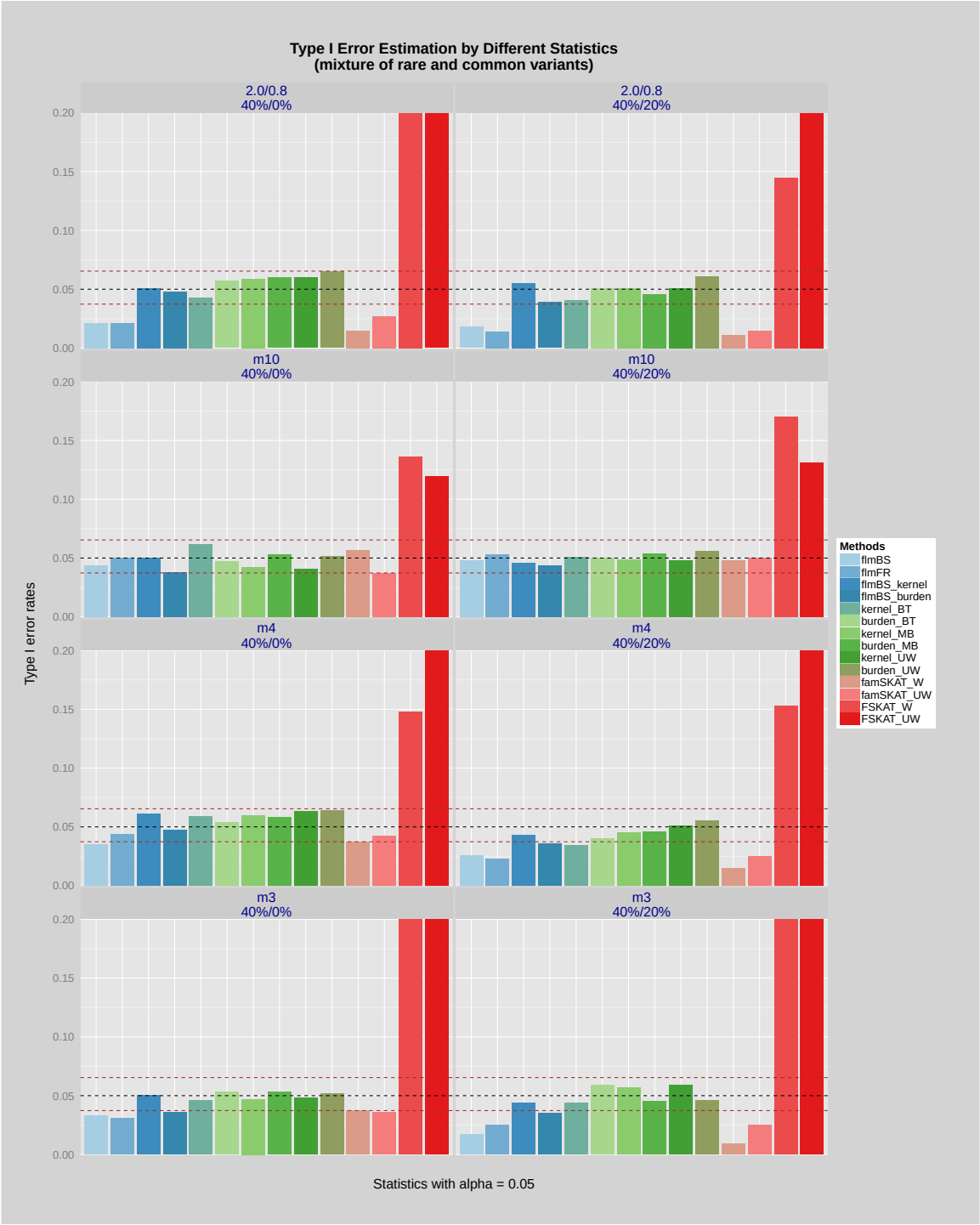

Figure A.5: Type I error rates of a combination of common and rare risk variants ($\alpha = 0.05$, a re-sampled set of 40% risk variants). Type I error rates for a re-sampled set of 40% combined common and rare risk variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

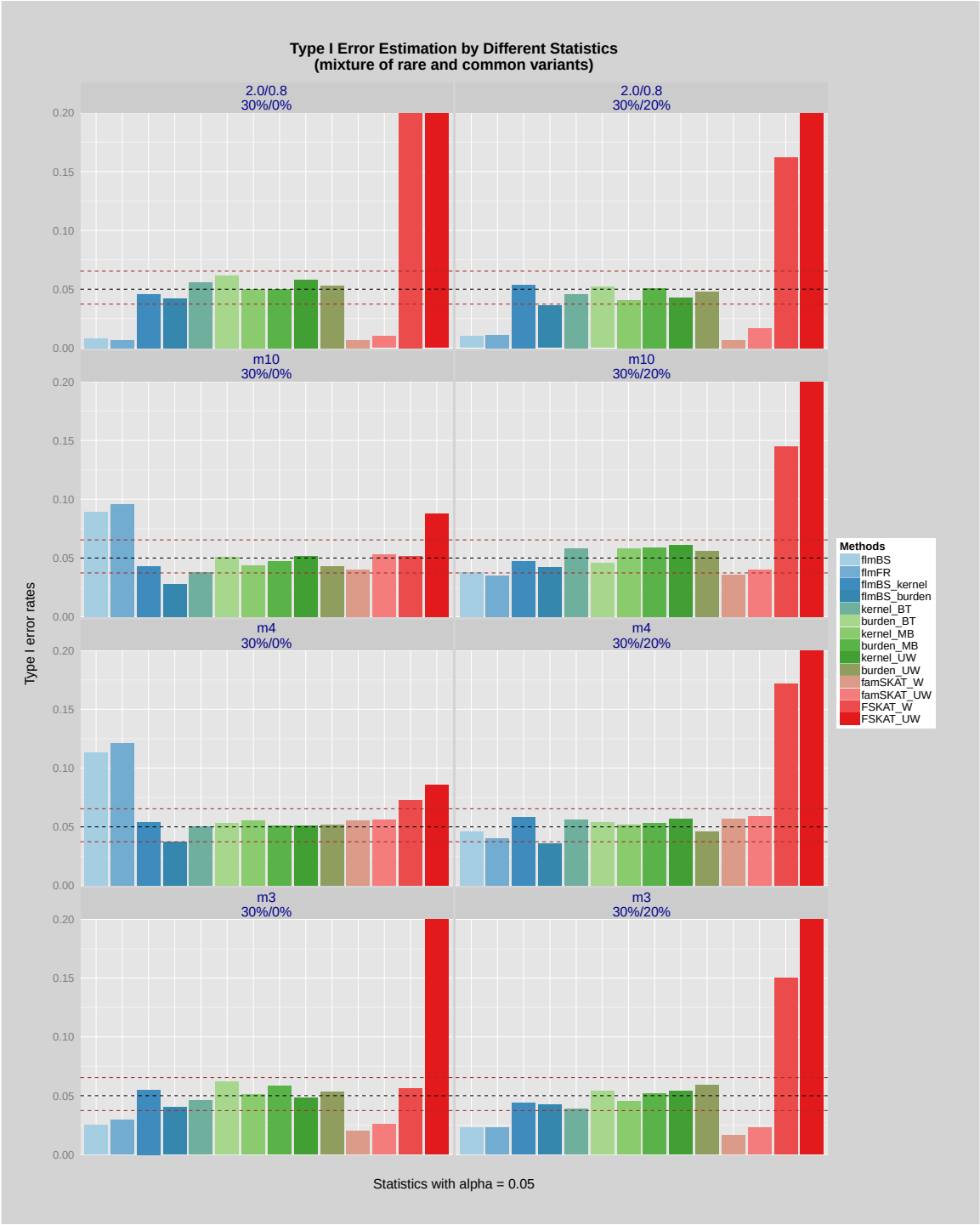

Figure A.6: Type I error rates of a combination of common and rare risk variants ($\alpha = 0.05$, 30% risk variants). Type I error rates for 30% rare risk variants (a subset of those in Figure A.4) with or without a mixture of 20% protective variants (the same ones as in Figure A.4). Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. Notations of statistics and scenarios are defined in Table 8.

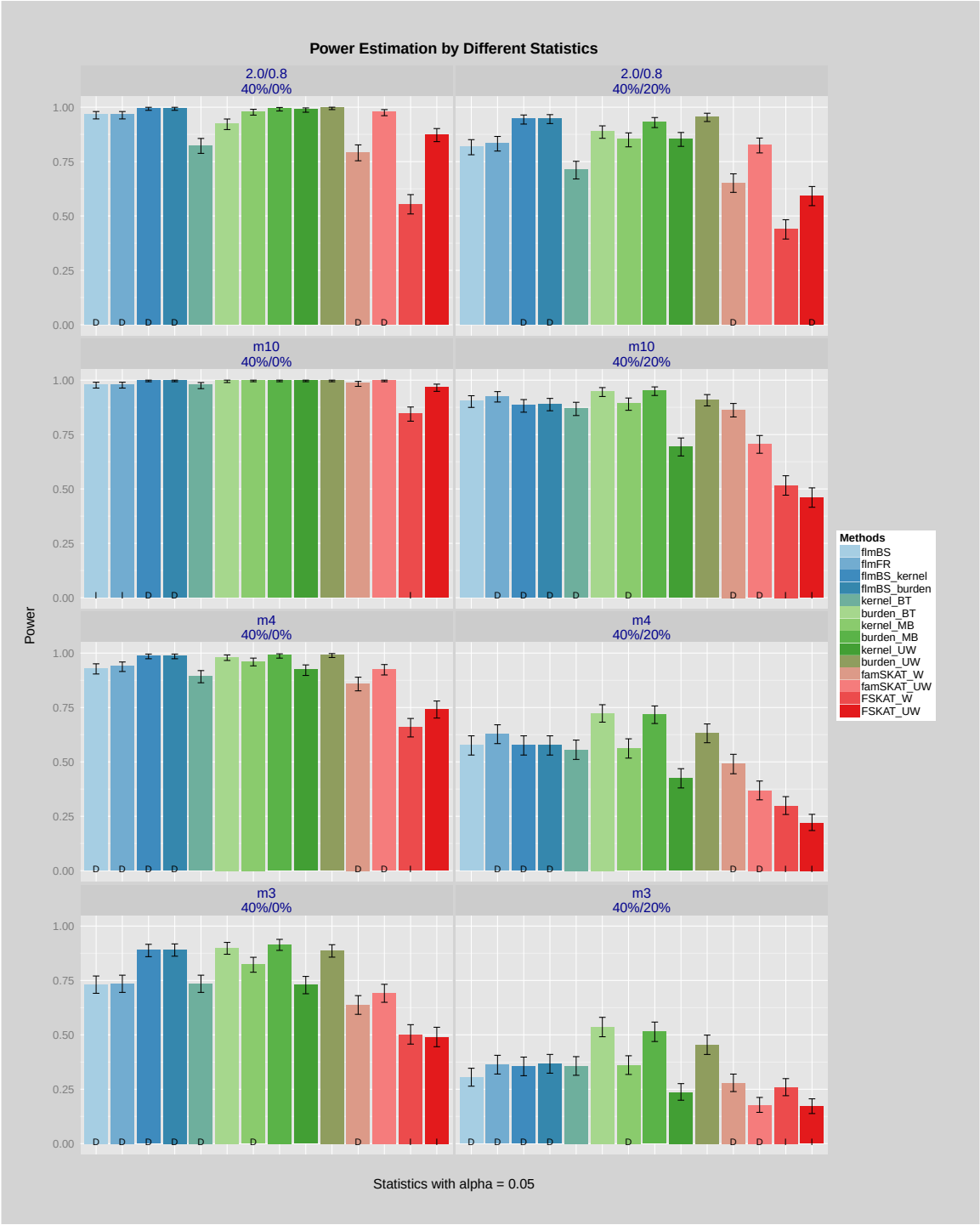

Figure A.7: Power levels of rare risk variants ($\alpha = 0.05$, 40% risk variants). Power levels for 40% rare risk variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. "I" or "D" at the bottom of each bar indicates the inflation or deflation of its corresponding type I error rate. Notations of statistics and scenarios are defined in Table 8.

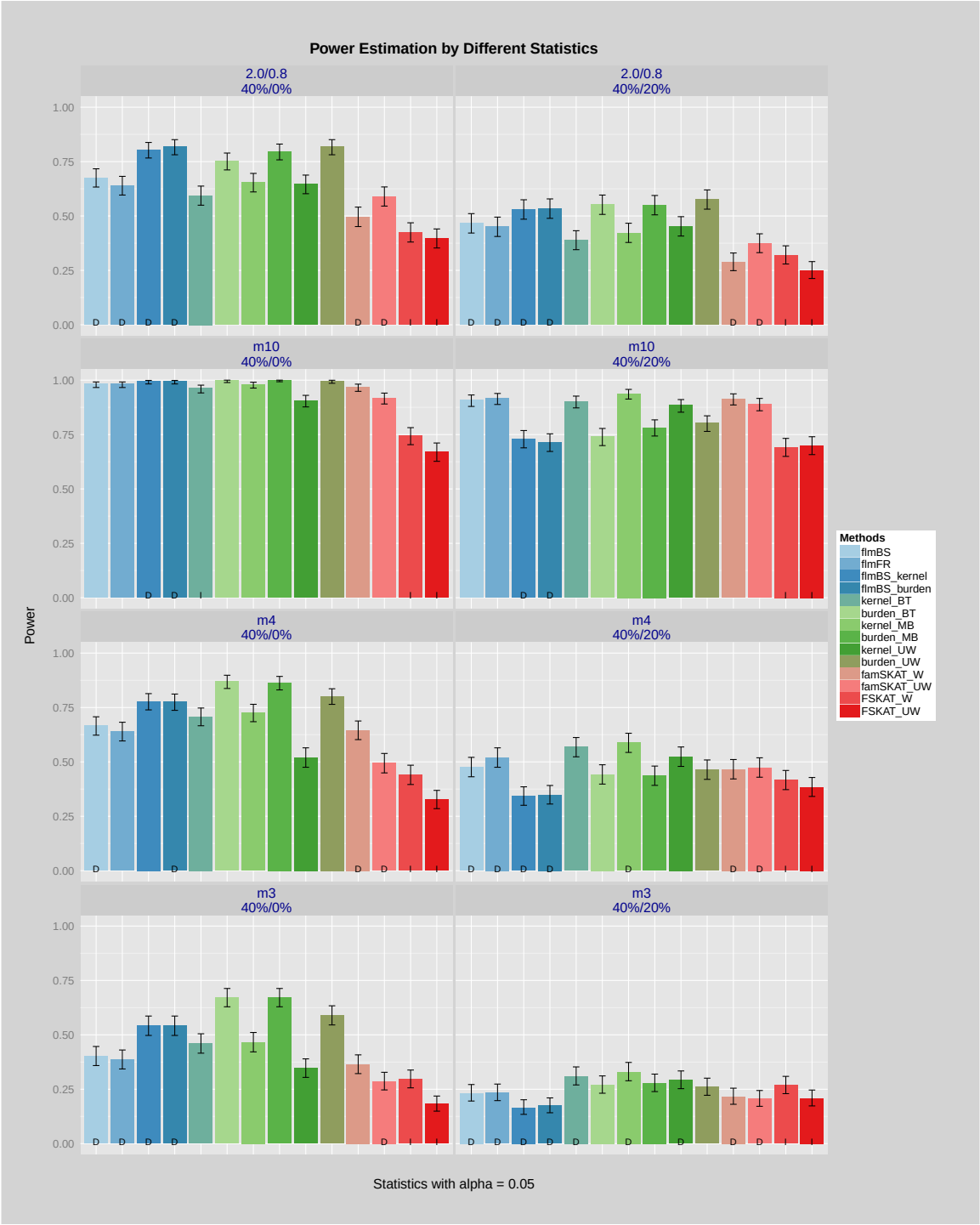

Figure A.8: Power levels of rare risk variants ($\alpha = 0.05$, a re-sampled set of 40% risk variants). Power levels for a re-sampled set of 40% rare risk variants with or without a mixture of 20% protective variants. Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. "I" or "D" at the bottom of each bar indicates the inflation or deflation of its corresponding type I error rate. Notations of statistics and scenarios are defined in Table 8.

Figure A.9: Power levels of rare risk variants ($\alpha = 0.05$, 30% risk variants). Power levels for 30% rare risk variants (a subset of those in Figure A.7) with or without a mixture of 20% protective variants (the same ones as in Figure A.7). Nominal type I error rate is 0.05. OR for risk variants is 2.0 or MAF-dependent, and OR for protective variants is 0.8 or MAF-dependent. "I" or "D" at the bottom of each bar indicates the inflation or deflation of its corresponding type I error rate. Notations of statistics and scenarios are defined in Table 8.

APPENDIX B

R CODES: SIMULATION STUDIES AND REAL DATA ANALYSES

B.1 GENERATE PEDIGREE STRUCTURE

We wrote a function called "simuPed" in R to generate related individuals with known pedigree structures for the simulation study in Section 3.1.6. The function took two arguments: *Nped* specified number of pedigrees we desired to generate, and *Maxsib* specified the maximum siblings within a certain generation.

```
#### Simulate pedigree structure
#### Compute kinship coefficients
#### Generate a seed file including 1,000 numbers
#### Clean the haplotype data

rm(list=ls())
require(kinship2)
DIR <- "/Volumes/Macintosh HD 2/Dissertation/Programs/Simulation/"
setwd(DIR)
kinMatName <- "simu_chol_cor.dat"
ITER <- 1000      # number of iterations
snpS <- 1000      # starting point of a gene
snpL <- 14*1000   # median 14 kb, mean 27 kb
RARE <- 0.05      # filter for rare variants
CAUSAL <- 0.4     # percent of the variants that are deleterious
PROTECT <- 0.2    # percent of the variants that are protective
ORrisk <- 2.0     # risk variants
ORprotect <- 0.8  # protective variants

############################################################
## write a function to generate pedigree structure
############################################################

simuPed <- function(Nped, Maxsib= 7) {
  if (missing(Nped) | Nped %% 1 !=0)
    stop("missing or an invalid number of pedigrees")
  if (missing(Maxsib)| Maxsib %% 1 !=0 | Maxsib < 2)
    stop("missing or an invalid number of sibling size")
```

```
  if (Maxsib >= 15)
    stop("sibling size too large")
  if (Nped >= 1500)
    warning("specify a large number of pedigrees or sibling size, simulation may be slow",
        immediate.= TRUE)

  ### parameters input for simulations ###
  N <- Nped          # number of pedigrees, must be greater than 1 #
  Nparent <- 2*N     # number of parents #
  NEXTProb <- 1/4    # proportion of the third generations #

  ### determine the progeny size by a negative distribution ###
  r <- c(2:Maxsib); n <- 2.84; P <- 0.93; Q <- 1+P
  Pr <- choose(n+r-1,r)*(P/Q)^r*Q^(-n)
  SIBSIZE <- matrix(c(r,Pr), byrow= TRUE, nrow= 2)

  ### choose sibship size ###
  sibsize <- sample(SIBSIZE[1, ], N, replace= TRUE, prob= SIBSIZE[2, ])
  pedsize <- sibsize + 2    # add in parents #

  ### generate data under simulation model ###
  Ped <- c(rep(1:N,pedsize))
  Per <- unlist(sapply(pedsize,seq))
  simuData <- data.frame(cbind(Ped,Per))
  colnames(simuData) <- c("Ped","Per")
  NN <- nrow(simuData)  # total number of subjects #
  simuData$Father <- 1
  simuData$Mother <- 2
  simuData$Sex <- 1                          # male #
  simuData[simuData$Per == 2, "Sex"] <- 2    # female founder #
  simuData[simuData$Per == 1 | simuData$Per == 2, "Father"] <- 0
  simuData[simuData$Per == 1 | simuData$Per == 2, "Mother"] <- 0
  simuData[simuData$Per != 1 & simuData$Per != 2, "Sex"] <- sample(c(1,2), NN-Nped*2,
      replace = TRUE)

  ### generate the third generations ###
  simuData$index <- 1:NN
  subFamID <- simuData[simuData$Father != 0, "index"]
  subFamID <- sample(subFamID, size= floor(length(subFamID)*NEXTProb), replace= FALSE)
  if (length(subFamID) != 0) {
    nextGen <- simuData[simuData$index %in% subFamID, ]
    nextGen$couple <- (NN+1) : (NN+1+nrow(nextGen)-1)
    nextGen$couple <- as.numeric(paste0("2",nextGen$couple))
    nextGen$Sex2 <- ifelse(nextGen$Sex == 1, 2, 1)
    nextGen1 <- nextGen[ ,c("Ped","Per","Father","Mother","Sex")]
    nextGen2 <- nextGen[ ,c("Ped","couple","Sex2")]
    nextGen2$Father <- nextGen2$Mother <- 0
    colnames(nextGen2) <- c("Ped","Per","Sex","Father","Mother")
    simuData2 <- rbind(nextGen1,nextGen2)
    simuData2$matchID <- rep(1:nrow(nextGen1),times= 2)
    simuData2 <- simuData2[order(simuData2$matchID,simuData2$Per), ]

    N2 <- length(unique(simuData2$matchID))
    sibsize2 <- sample(SIBSIZE[1, ], N2, replace= TRUE, prob= SIBSIZE[2, ])
    matchID <- c(rep(unique(simuData2$matchID),sibsize2))
    Per2 <- unlist(sapply(sibsize2, seq))
    Per2 <- ( NN+nrow(nextGen2) ) : ( NN+nrow(nextGen2)+length(matchID)-1 )
    Per2 <- as.numeric(paste0("3",Per2))
    gen3Data <- data.frame(cbind(matchID,Per2))

    gen3Data$Ped <- 0
    gen3Data$Father <- gen3Data$Mother <- -1
    gen3Data$Sex <- sample(c(1,2), nrow(gen3Data), replace= TRUE)
    colnames(gen3Data)[colnames(gen3Data) == "Per2"] <- "Per"
    simuData2 <- rbind(simuData2, gen3Data)
    simuData2 <- simuData2[order(simuData2$matchID), ]

    for (j in matchID) {
```

```
      tmp <- simuData2[simuData2$matchID == j, ]
      tmp <- tmp[1:2, ]
      ped <- tmp[1,"Ped"]
      father <- tmp[tmp$Sex == 1,"Per"]
      mother <- tmp[tmp$Sex == 2, "Per"]
      simuData2[simuData2$matchID == j & simuData2$Ped == 0, "Ped"] <- ped
      simuData2[simuData2$matchID == j & simuData2$Father == -1, "Father"] <- father
      simuData2[simuData2$matchID == j & simuData2$Mother == -1, "Mother"] <- mother
    }

    simuData2$matchID <- NULL ; simuData$index <- NULL

    simuData <- rbind(simuData,simuData2)
    simuData <- simuData[!duplicated(simuData), ]
    simuData <- simuData[order(simuData$Ped,simuData$Per), ]
  } else {
    simuData$index <- NULL
  }
  return (simuData)
}


############################################################
## generate the pedigree structure and write out
############################################################

SEED <- 25
set.seed(SEED)
pedstr <- simuPed(Nped=  25, Maxsib= 7)
pedstr$ID <- 1:nrow(pedstr)
save(pedstr, file= "pedstr.RData")


############################################################
## compute kinship coefficients and write out
############################################################

kinObj <- pedigree(id= pedstr$Per, dadid= pedstr$Father, momid= pedstr$Mother, sex=
    pedstr$Sex, famid= pedstr$Ped)
kinMatrix <- kinship(kinObj)
S <- 2*kinMatrix
S <- as.matrix(S)
save(S, file= "kinCoef.RData")
cholS <- chol(S)  # cholesky decomposition of kinship matrix #
cholS <- t(cholS) # lower triangle matrix #
colnames(cholS) <- rownames(cholS) <- pedstr$ID
write.table(cholS,kinMatName,sep= "\t",quote= FALSE,col.names= TRUE,row.names= TRUE)


############################################################
## generate a seed file including 1000 numbers
############################################################
seedfile <- sample.int(1e8, ITER, replace= FALSE)
save(seedfile, file= "seedFile.RData")


############################################################
## clean the haplotype data generated by Qi
############################################################

## read in haplotype data simulated by Qi ##
hap <- read.table("Haplotype.txt", header= FALSE, sep= " ", stringsAsFactors= FALSE)
snp <- read.table("SNPInfo.txt", header= TRUE, sep= " ", stringsAsFactors= TRUE)
subsnp <- snp
subsnp$rs <- paste0("rs",subsnp$rs)
colnames(hap) <- subsnp$rs

## get a subregion with a length of 14 kb, and save the subsets ##
```

```
subsnp <- subsnp[subsnp$position >= snpS & subsnp$position <= snpS+snpL, ]
subhap <- hap[ ,subsnp$rs]

## filter out common variants ##
maf <- colMeans(subhap)
subsnp <- subsnp[maf > 0 & maf < RARE, ]
subhap <- subhap[ ,subsnp$rs]

save(subhap, file= "Haplotype_sub.RData")
save(subsnp, file= "SNPInfo_sub.RData")


############################################################
## clean the haplotype data generated by Qi
############################################################

## sample ~% causal variants ##
NRhap <- nrow(subhap)
NChap <- ncol(subhap)
Ncausal <- ceiling(CAUSAL*NChap)
Nprotect <- ceiling(PROTECT*NChap)

## 1. 40 % causative with fixed OR ##
logORrisk1 <- log(c(rep(ORrisk,Ncausal),rep(1,NChap-Ncausal)))
logORrisk1 <- sample(logORrisk1)
save(logORrisk1, file= "logOR_2_40.RData")

## 2. 40 % causative 20 % protective with fixed OR ##
logORrisk2 <- logORrisk1
index0 <- which(logORrisk1 == 0)
indexRisk <- which(logORrisk1 > 0)
sampleIndex0 <- sample(index0, Nprotect, replace= FALSE)
logORrisk2[sampleIndex0] <- log(ORprotect)
save(logORrisk2, file= "logOR_2_40_08_20.RData")

## 3. 40 % causative with MAF dependent OR ##
submaf <- colMeans(subhap)
submaf <- as.vector(submaf)
#-----------------------------------------------------------#
logORmaf <- log(10)/4 * abs(log10(submaf))
logORrisk3 <- logORmaf
logORrisk3[index0] <- 0
save(logORrisk3, file= "logOR_m_40.RData")

logORmaf3 <- log(3)/4 * abs(log10(submaf))
logORrisk3_m3 <- logORmaf3
logORrisk3_m3[index0] <- 0
save(logORrisk3_m3, file= "logOR_m3_40.RData")

logORmaf4 <- log(4)/4 * abs(log10(submaf))
logORrisk3_m4 <- logORmaf4
logORrisk3_m4[index0] <- 0
save(logORrisk3_m4, file= "logOR_m4_40.RData")

#logORmaf5 <- log(5)/4 * abs(log10(submaf))
#logORrisk4_m5 <- logORmaf5
#logORrisk4_m5[index0] <- 0
#save(logORrisk4_m5, file= "logOR_m5_40.RData")
#-----------------------------------------------------------#

## 4. 40 % causative 20 % protective with MAF dependent OR ##
#-----------------------------------------------------------#
logORrisk4 <- logORrisk2
indexPos <- which(logORrisk2 > 0)
logORrisk4[indexPos] <- logORmaf[indexPos]
indexNeg <- which(logORrisk2 < 0)
logORrisk4[indexNeg] <- -logORmaf[indexNeg]
save(logORrisk4, file= "logOR_m_40_m_20.RData")
```

```
230
231 logORrisk4_m3 <- logORrisk2
232 indexPos <- which(logORrisk2 > 0)
233 logORrisk4_m3[indexPos] <- logORmaf3[indexPos]
234 indexNeg <- which(logORrisk2 < 0)
235 logORrisk4_m3[indexNeg] <- -logORmaf3[indexNeg]
236 save(logORrisk4_m3, file= "logOR_m3_40_m3_20.RData")
237
238 logORrisk4_m4 <- logORrisk2
239 indexPos <- which(logORrisk2 > 0)
240 logORrisk4_m4[indexPos] <- logORmaf4[indexPos]
241 indexNeg <- which(logORrisk2 < 0)
242 logORrisk4_m4[indexNeg] <- -logORmaf4[indexNeg]
243 save(logORrisk4_m4, file= "logOR_m4_40_m4_20.RData")
244
245 #------------------------------------------------------------#
246
247 #------------------------------------------------------------#
248
249 ## select 1/4 causative variants from 40 % ones ##
250 Nnull <- floor(1/4 * Ncausal)
251 nullVariant <- sample(indexRisk, Nnull)
252
253 ## 5. 30 % causative with fixed OR, subset ##
254 logORrisk5 <- logORrisk1
255 logORrisk5[nullVariant] <- 0
256 save(logORrisk5, file= "logOR_2_30.RData")
257
258 ## 6. 30 % causative 20 % protective with fixed OR, subset ##
259 logORrisk6 <- logORrisk2
260 logORrisk6[nullVariant] <- 0
261 save(logORrisk6, file= "logOR_2_30_08_20.RData")
262
263 ## 7. 30 % causative with MAF dependent OR, subset ##
264 logORrisk7 <- logORrisk3
265 logORrisk7[nullVariant] <- 0
266 save(logORrisk7, file= "logOR_m_30.RData")
267
268 logORrisk7_m3 <- logORrisk3_m3
269 logORrisk7_m3[nullVariant] <- 0
270 save(logORrisk7_m3, file= "logOR_m3_30.RData")
271
272 logORrisk7_m4 <- logORrisk3_m4
273 logORrisk7_m4[nullVariant] <- 0
274 save(logORrisk7_m4, file= "logOR_m4_30.RData")
275
276 ## 8. 30 % causative 20 % protective with MAF dependent OR, subset ##
277 logORrisk8 <- logORrisk4
278 logORrisk8[nullVariant] <- 0
279 save(logORrisk8, file= "logOR_m_30_m_20.RData")
280
281 logORrisk8_m3 <- logORrisk4_m3
282 logORrisk8_m3[nullVariant] <- 0
283 save(logORrisk8_m3, file= "logOR_m3_30_m3_20.RData")
284
285 logORrisk8_m4 <- logORrisk4_m4
286 logORrisk8_m4[nullVariant] <- 0
287 save(logORrisk8_m4, file= "logOR_m4_30_m4_20.RData")
288
289 #------------------------------------------------------------#
290
291 ## 9. Simulate an uncorrelated data set  ##
292 Ncausal <- length(which(logORrisk1 > 0))
293 logORrisk9 <- log(c(rep(ORrisk,Ncausal),rep(1,NChap-Ncausal)))
294 logORrisk9 <- sample(logORrisk9)
295 save(logORrisk9, file= "logOR_2_40_new.RData")
296
297 ## 10. 40 % causative 20 % protective with fixed OR ##
```

```
logORrisk10 <- logORrisk9
index0 <- which(logORrisk9 == 0)
sampleIndex0 <- sample(index0, Nprotect, replace= FALSE)
logORrisk10[sampleIndex0] <- log(ORprotect)
save(logORrisk10, file= "logOR_2_40_08_20_new.RData")

## 11. 40 % causative with MAF dependent OR ##
submaf <- colMeans(subhap)
submaf <- as.vector(submaf)
logORmaf10 <- log(10)/4 * abs(log10(submaf))
logORrisk11_m10 <- logORmaf10
logORrisk11_m10[index0] <- 0
save(logORrisk11_m10, file= "logOR_m_40_new.RData")

logORmaf4 <- log(4)/4 * abs(log10(submaf))
logORrisk11_m4 <- logORmaf4
logORrisk11_m4[index0] <- 0
save(logORrisk11_m4, file= "logOR_m4_40_new.RData")

logORmaf3 <- log(3)/4 * abs(log10(submaf))
logORrisk11_m3 <- logORmaf3
logORrisk11_m3[index0] <- 0
save(logORrisk11_m3, file= "logOR_m3_40_new.RData")

## 12. 40 % causative 20 % protective with MAF dependent OR ##
logORrisk12_m10 <- logORrisk10
indexPos <- which(logORrisk10 > 0)
logORrisk12_m10[indexPos] <- logORmaf10[indexPos]
indexNeg <- which(logORrisk10 < 0)
logORrisk12_m10[indexNeg] <- -logORmaf10[indexNeg]
save(logORrisk12_m10, file= "logOR_m_40_m_20_new.RData")

logORrisk12_m4 <- logORrisk10
indexPos <- which(logORrisk10 > 0)
logORrisk12_m4[indexPos] <- logORmaf4[indexPos]
indexNeg <- which(logORrisk10 < 0)
logORrisk12_m4[indexNeg] <- -logORmaf4[indexNeg]
save(logORrisk12_m4, file= "logOR_m4_40_m4_20_new.RData")

logORrisk12_m3 <- logORrisk10
indexPos <- which(logORrisk10 > 0)
logORrisk12_m3[indexPos] <- logORmaf3[indexPos]
indexNeg <- which(logORrisk10 < 0)
logORrisk12_m3[indexNeg] <- -logORmaf3[indexNeg]
save(logORrisk12_m3, file= "logOR_m3_40_m3_20_new.RData")
```

B.2 SIMULATE GENOTYPES AND ASSIGN TRAITS

We ran the following program on a high performance computing cluster to simulate genotype data in Section 3.1.6. For each iteration, it assigned haplotypes to the founders in a pedigree, transmitted to offspring, converted to genotypes, and smoothed the data by invoking basis functions. It also assigned traits to all simulated individuals and generated input files for

the GLOGS program.

```
#### This program will generate smoothed genotype data for type I error and power
    estimations.

rm(list=ls())

args= (commandArgs(TRUE))
if (length(args) == 0) {
  print("No arguments supplied.")
  seedIndex = 1
} else {
  for (i in 1:length(args)) {
    eval(parse(text=args[[i]]))
  }
}

RpkgDIR <- "/home/dweeks/yij5/Dissertation/Simulation/Rpackage/"
require("MASS",lib.loc= RpkgDIR)
require("fda",lib.loc= RpkgDIR)

DIRcurrent <- paste0("/home/dweeks/yij5/Dissertation/Simulation/Iterations/iter_",seedIndex,
    "/")
DIRinput <- "/home/dweeks/yij5/Dissertation/Simulation/Input/"
DIRoutput <- "/home/dweeks/yij5/Dissertation/Simulation/Output/"

## define file names ##
phenoName <- "simu_data.dat"
uMatFile <- "u_matrix_star.dat"
ORfileName <- "logOR_m3_40_new.RData"

## define parameters ##
PREV <- 0.05   # disease prevalence
POLYGEN <- 0.2 # polygenic effect

order <- 4
basis <- 5   # number of basis functions
beta0 <- log(PREV/(1 - PREV))   # intercept adjusting population prevalence

################################################################
## Read in the haplotype data and seed file
## Sample causal genetic variants
################################################################

## read in haplotype data simulatd by Qi ##
setwd(DIRinput)
hap <- get(load("Haplotype_sub.RData"))
snp <- get(load("SNPInfo_sub.RData"))
kinCoef <- get(load("kinCoef.RData"))
seeds <- get(load("seedFile.RData"))
SEED <- seeds[seedIndex]
set.seed(SEED)

## sample causal variants ##
NRhap <- nrow(hap)
NChap <- ncol(hap)
logORrisk <- get(load(ORfileName))

################################################################
## Simulate and acertain 25 pedigrees with at least 2 cases.
## Transmit haplotypes to offspring.
## Convert haplotypes to genotypes, and assign phenotypes.
################################################################

## read in the pedigree structure ##
pedstr <- get(load("pedstr.RData"))
pedstr$trait <- -1    # preassign the trait #
family <- unique(pedstr$Ped)
```

```
65
66 ## identify pedigree "founders" and sample haplotypes ##
67 pedstr$m2 <- pedstr$m1 <- 0
68 #founderID <- pedstr[pedstr$Father == 0, "ID"]
69 #Nfounder <- length(founderID)
70 #pedstr[pedstr$ID %in% founderID, "m1"] <- sample(c(1:NRhap), Nfounder, replace= TRUE)
71 #pedstr[pedstr$ID %in% founderID, "m2"] <- sample(c(1:NRhap), Nfounder, replace= TRUE)
72
73 ## assign haplotypes and genotypes ##
74 powerPed  <- data.frame()
75 powerGeno <- data.frame()
76 NCgeno <- NChap
77
78 system.time(
79 for (f in family) {
80   substr <- pedstr[pedstr$Ped == f, ]
81   founderID <- substr[substr$Father == 0, "ID"]
82   Nfounder <- length(founderID)
83   nonFounderID <- substr[substr$Father != 0, "ID"]
84   index <- 0    # ascertainment index #
85
86   while (index == 0) {
87     ## sample haplotypes for founders ##
88     substr[substr$ID %in% founderID, "m1"] <- sample(c(1:NRhap), Nfounder, replace= TRUE)
89     substr[substr$ID %in% founderID, "m2"] <- sample(c(1:NRhap), Nfounder, replace= TRUE)
90
91     ## sample haplotypes and convert to genotypes ##
92     for (j in nonFounderID) {
93       fatherPer <- substr[substr$ID == j, "Father"]
94       fatherPool <- as.vector(substr[substr$Per == fatherPer, c("m1","m2")])
95       substr[substr$ID == j, "m1"] <- sample(fatherPool, 1, replace= TRUE)
96       motherPer <- substr[substr$ID == j, "Mother"]
97       motherPool <- as.vector(substr[substr$Per == motherPer, c("m1","m2")])
98       substr[substr$ID == j, "m2"] <- sample(motherPool, 1, replace= TRUE)
99     }
100
101    NRsubstr <- nrow(substr)
102    geno <- hap[substr$m1, ] + hap[substr$m2, ]
103    colnames(geno) <- colnames(hap)
104
105    ## simulate polygenic effect ##
106    muMean <- rep(0, NRsubstr)
107    kinStart <- substr[1,"ID"]
108    kinEnd <- substr[NRsubstr,"ID"]
109    corSigma <- kinCoef[kinStart:kinEnd,kinStart:kinEnd]
110    a <- mvrnorm(n= 1, mu= muMean, Sigma= corSigma, tol= 1e-16, empirical= FALSE, EISPACK=
           FALSE)
111
112    ## compute the Bernoulli probability of affected status conditional on genotype ##
113    ## based on Schaid's proposed model ##
114    for (k in 1:NRsubstr) {
115      g <- geno[k, ]
116      A <- beta0 + sum(logORrisk*g) + POLYGEN*a[k]
117      mu <- exp(A) / (1 + exp(A))
118      substr[k,"trait"] <- sample(c(1,0), 1, replace= FALSE, prob= c(mu,1-mu))
119    }
120
121    ## Ascertain at least 2 affected siblings in the pedigree ##
122    affectMomID <- substr[substr$trait == 1, "Mother"]
123    dup <- affectMomID[duplicated(affectMomID)]
124    if (sum(dup) != 0) {
125      powerPed  <- rbind(powerPed, substr)
126      powerGeno <- rbind(powerGeno, geno)
127      index <- 1
128    }
129  } # end of while loop #
130 }  # end of for loop #
131 )
```

```
132
133 powerPed$m1 <- powerPed$m2 <- NULL
134
135 ###################################################################
136 ## Compute MAF, and keep the markers in (0, 0.05)
137 ###################################################################
138
139 ## remove non-polymorphic variants ##
140 maf <- colSums(powerGeno)
141 powerGeno <- powerGeno[ ,maf > 0]
142 # add in if condition here: no markers selected #
143
144 ###################################################################
145 ## Invoke the GLOGS program in R.
146 ###################################################################
147
148 ## input files for GLOGS ##
149 setwd(DIRcurrent)
150 phenoCov <- powerPed[ ,c("ID","trait")]
151 write.table(phenoCov,phenoName,sep= "\t",quote= FALSE,col.names= FALSE,row.names= FALSE)
152
153 ## invoke GLOGS in R ##
154 ## estimate initial non-genotype covariate effect ##
155 NRpowerPed <- nrow(powerPed)
156 glmEst <- glm(formula= powerPed$trait ~1, family= "binomial")
157 initial <- summary(glmEst)$coefficients[1]
158
159 ###################################################################
160 ## Weight the variants by the beta-smooth only model
161 ###################################################################
162
163 pos <- snp[snp$rs %in% colnames(powerGeno),"position"] # already sorted #
164 nsample <- nrow(powerGeno)
165 nsnp    <- ncol(powerGeno)
166
167 ## normalize position ##
168 if (max(pos) > 1) {
169   pos <- (pos - min(pos)) / (max(pos) - min(pos))
170 }
171
172 ## generate basis functions ##
173 betabasis <- create.bspline.basis(norder= order, nbasis= basis)
174 fourierbasis <- create.fourier.basis(c(0,1), nbasis= basis)
175
176 B <- eval.basis(pos, betabasis)
177 FR <- eval.basis(pos, fourierbasis)
178
179 powerGeno <- as.matrix(powerGeno)
180
181 UJ_power <- powerGeno %*% B
182 UJ_power_FR <- powerGeno %*% FR
183
184 ###################################################################
185 ## Output the pedigree structure with simulated trait.
186 ## Output the two unweighted genotype files.
187 ###################################################################
188
189 save(powerGeno, file= paste0("powerGeno_", seedIndex, ".RData"))
190
191 save(UJ_power,    file= paste0("UJ_power_BS_", seedIndex, ".RData"))
192 save(UJ_power_FR, file= paste0("UJ_power_FR_", seedIndex, ".RData"))
193
194 write.table(initial, file= paste0("initial_", seedIndex, ".txt"),row.names= FALSE, col.names
        = FALSE, quote= FALSE)
195 save(powerPed, file= paste0("pedstrTrait_", seedIndex, ".RData"))
```

B.3 COMPUTE STATISTICS AND P-VALUES IN THE SIMULATION STUDY

B.3.1 GFLMM

We ran the following program on a high performance computing cluster to estimate type I error rates and power levels in Section 3.1.6. For each iteration, it read in the pedigree and genotype data, approximated the integral by using Sobol points produced by the GLOGS program, computed Rao's score test statistics, and estimated and output p-values.

```
#### This program will estimate type I error rates and power levels for GFLMM.

rm(list=ls())

args= (commandArgs(TRUE))
if (length(args) == 0) {
  print("No arguments supplied.")
  seedIndex = 1
} else {
  for (i in 1:length(args)) {
    eval(parse(text=args[[i]]))
  }
}

DIRcurrent <- paste0("/home/dweeks/yij5/Dissertation/Simulation/Iterations/iter_",seedIndex,
    "/")
DIRinput <- "/home/dweeks/yij5/Dissertation/Simulation/Input"
DIRoutput <- "/home/dweeks/yij5/Dissertation/Simulation/Output/"
RpkgDIR <- "/home/dweeks/yij5/Dissertation/Simulation/Rpackage/"

require("MASS", lib.loc= RpkgDIR)

## define file names ##
phenoName <- "simu_data.dat"
uMatFile <- "u_matrix_star.dat"

## define parameters ##
Z <- 1          # covariate vector
C <- 400000     # number of cubature sizes
W <- 1/C        # equal weights

setwd(DIRinput)
uMat <- scan(uMatFile, sep= "",quiet= TRUE)

################################################################
## Read in the genotype data for analysis.
################################################################

setwd(DIRcurrent)

Gpower    <- get(load(paste0("UJ_power_BS_", seedIndex, ".RData")))
Gpower_FR <- get(load(paste0("UJ_power_FR_", seedIndex, ".RData")))

phenoCov <- read.table(phenoName, header= FALSE, sep= "\t", stringsAsFactors= FALSE)
colnames(phenoCov) <- c("ID","trait")
NRpowerPed <- nrow(phenoCov)


```

```
################################################################
## Get the final estimated parameters and compute p-value.
################################################################

#rmCMD <- paste("rm -f", uMatFile, sep= " ")
#system(rmCMD, wait= FALSE)

estParameters <- scan("estParameters.txt", quiet= TRUE)
estBetaNull <- estParameters[1]
estSigma    <- estParameters[2]
lnL         <- estParameters[3]

y <- phenoCov$trait

numG_power    <- ncol(Gpower)     # number of variants #
numG_power_FR <- ncol(Gpower_FR)

Gpower    <- as.data.frame(Gpower)
Gpower_FR <- as.data.frame(Gpower_FR )

Gpower$Z <- 1
Gpower_FR$Z <- 1

S_theta_numerator_power <- S_theta_numerator_power_FR <- 0
I_theta_numerator_power <- I_theta_numerator_power_FR <- 0

S_theta_denorminator <- exp(lnL-log(W))
I_theta_denorminator <- S_theta_denorminator

for (c in 1:C) {
  ## compute p ##
  u_vec <- uMat[((c-1)*NRpowerPed+1):(c*NRpowerPed)]
  B <- estBetaNull+estSigma*u_vec
  p <- exp(B)/(1+exp(B))

  ## compute likelihood given c: exp(l_a_c) ##
  l_c <- sum(y*log(p)+(1-y)*log(1-p))
  exp_l_c <- exp(l_c)

  Gpower$u_vec    <- u_vec
  Gpower_FR$u_vec <- u_vec

  t_z_theta_power    <- as.matrix(Gpower)
  t_z_theta_power_FR <- as.matrix(Gpower_FR)

  ## compute z= (G,1,a(n))' and Dl_theta= sum((y-p)*z) ## #
  y_p <- matrix((y-p), nrow= 1, ncol= NRpowerPed, byrow= TRUE)
  Dl_theta_power    <- (y_p) %*% t_z_theta_power
  Dl_theta_power_FR <- (y_p) %*% t_z_theta_power_FR

  ## assume equal weights ##
  S_theta_numerator_power    <- S_theta_numerator_power    + Dl_theta_power*exp_l_c
  S_theta_numerator_power_FR <- S_theta_numerator_power_FR + Dl_theta_power_FR*exp_l_c

  ## D2l_theta= sum((p^2-p)*z*z') ##
  z_theta_power    <- t(t_z_theta_power)
  z_theta_power_FR <- t(t_z_theta_power_FR)

  row_z_theta_power    <- nrow(z_theta_power)
  row_z_theta_power_FR <- nrow(z_theta_power_FR)

  p2_p <- p^2-p

  p2_p_power    <- matrix(rep(p2_p,row_z_theta_power), nrow= row_z_theta_power, ncol=
      NRpowerPed, byrow= TRUE)
  p2_p_power_FR <- matrix(rep(p2_p,row_z_theta_power_FR), nrow= row_z_theta_power_FR, ncol=
      NRpowerPed, byrow= TRUE)

```

```
  D2l_theta_power    <- (p2_p*z_theta_power) %*% t_z_theta_power
  D2l_theta_power_FR <- (p2_p*z_theta_power_FR) %*% t_z_theta_power_FR

  I_theta_numerator_power    <- I_theta_numerator_power + (D2l_theta_power + t(
      Dl_theta_power) %*% Dl_theta_power)*exp_l_c
  I_theta_numerator_power_FR <- I_theta_numerator_power_FR + (D2l_theta_power_FR + t(
      Dl_theta_power_FR) %*% Dl_theta_power_FR)*exp_l_c
}

## write a function to compute p-value of the score test ##
scoreP <- function(S_theta_numerator, I_theta_numerator, numG) {

  S_theta <- S_theta_numerator / S_theta_denorminator
  I_theta <- -(I_theta_numerator/I_theta_denorminator - t(S_theta) %*% S_theta)
  S_gamma <- S_theta[ ,1:numG, drop= FALSE]

  I_gamma_gamma <- I_theta[1:numG,1:numG, drop= FALSE]
  I_gamma_eta   <- I_theta[1:numG,(numG+1):ncol(I_theta), drop= FALSE]
  I_eta_gamma   <- I_theta[(numG+1):nrow(I_theta),1:numG, drop= FALSE]
  I_eta_eta     <- I_theta[(numG+1):nrow(I_theta),(numG+1):ncol(I_theta), drop= FALSE]
  inv_I_eta_eta <- ginv(I_eta_eta)
  I_gamma <- I_gamma_gamma- I_gamma_eta %*% inv_I_eta_eta %*% I_eta_gamma
  inv_I_gamma <- ginv(I_gamma)

  R <- S_gamma %*% inv_I_gamma %*% t(S_gamma)
  R <- c(R)
  pValue_R <- pchisq(R, df= numG, lower.tail= FALSE)

  return(pValue_R)
}

pValue_R_power    <- scoreP(S_theta_numerator_power,I_theta_numerator_power,numG_power)
pValue_R_power_FR <- scoreP(S_theta_numerator_power_FR,I_theta_numerator_power_FR,
    numG_power_FR)

## read out the results ##
results <- c(pValue_R_power,pValue_R_power_FR)
save(results, file= paste0("simu_result_", seedIndex ,".RData"))
```

B.3.2 Embedded approaches

The following program embedded FLM-smoothed genotypes in retrospective kernel-based approach and burden tests. the B-Spline basis was used to smooth the genotypes.

```
#### This program will embed FLM-smoothed genotypes in Schaid's kernel and burden tests

rm(list=ls())

args= (commandArgs(TRUE))
if (length(args) == 0) {
  print("No arguments supplied.")
  MODEL = "2_40"
} else {
  for (i in 1:length(args)) {
    eval(parse(text=args[[i]]))
  }
}

```

```
require(pedgene) ; require(fda)
order <- 4

#DIRmain <- "/Volumes/Macintosh HD 2/Dissertation/Programs/Simulation/otherStatistics/"
DIRmain <- "/Users/yjiang/Dissertation/Simulation/otherStatistics/"
#MODEL <- "m_40_m_20"
dataDIR <- paste0(DIRmain, "oriGenoPedtrait_", MODEL ,"/")
flmDIR <- paste0(DIRmain, "flmGenoSimuData_", MODEL, "/")
ORname <- paste0("logOR_", MODEL, ".RData")

#################################################################
## Read in simulated data sets.
#################################################################

setwd(DIRmain)
seeds <- get(load("seedFile.RData"))
snp <- get(load("SNPInfo_sub.RData"))
logORrisk <- get(load(ORname))
riskVariantList <- snp[which(logORrisk > 0), "rs"]
protectVariantList <- snp[which(logORrisk < 0), "rs"]

NFILES <- 500
for (f in 1:NFILES) {
  SEED <- seeds[f]
  powerGeno <- get(load(paste0(dataDIR,"powerGeno_",f,".RData")))
  pedstr <- get(load(paste0(dataDIR,"pedstrTrait_",f,".RData")))
  NCpowerGeno <- ncol(powerGeno)
  N <- nrow(pedstr)
  row.names(powerGeno) <- 1:N

  pos <- snp[snp$rs %in% colnames(powerGeno), "position"]
  if (max(pos) > 1) {
    pos <- (pos - min(pos)) / (max(pos) - min(pos))
  }
  ## generate basis functions ##
  if (NCpowerGeno > order) {     # number of non-polymorphic variants > 4 #
    betabasis <- create.bspline.basis(norder= order, nbasis= NCpowerGeno)
  } else {
    betabasis <- create.bspline.basis(norder= NCpowerGeno, nbasis= NCpowerGeno)
  }
  B <- eval.basis(pos, betabasis)
  powerGeno <- as.matrix(powerGeno)
  UJ <- powerGeno %*% B

  ################################################################
  ## Compute Schaid's kernel and burden statistics.
  ## pedgene package in R
  ################################################################

  weight1 <- rep(1,NCpowerGeno)
  schaidPed <- pedstr
  schaidPed$ID <- NULL
  colnames(schaidPed) <- c("ped","person","father","mother","sex","trait")
  pedPer <- schaidPed[ ,c("ped","person")]
  schaidPowerG <- cbind(pedPer,UJ)

  schaid_power <- pedgene(schaidPed, schaidPowerG, male.dose= 2, checkpeds= FALSE, weights=
      weight1, weights.mb= NULL, method= "kounen", acc.davies= 1e-9)
  pKernelPower <- schaid_power$pgdf$pval.kernel
  pBurdenPower <- schaid_power$pgdf$pval.burden

 ################################################################

  result <- c(pKernelPower, pBurdenPower,SEED)
  otherStats <- as.data.frame(t(result))
  outputFileName <- paste0("powerSchaidFLM_num_", MODEL, ".txt")
  write.table(otherStats, file= outputFileName, append= TRUE, quote= FALSE,
              sep= "\t", row.names= FALSE, col.names= FALSE)
```

```
}
```

B.3.3 Other statistics for comparison

The following program computed the statistics derived from the retrospective kernel and burden tests, famSKAT, and F-SKAT, which were used to compare with the GFLMM and the embedded approach developed in Chapter 3.

```
#### This program will compute other statistics for a comparison purpose.
####  Schaid's kernel and burden tests
####  famSKAT treating binary 0/1 as quantitative trait
####  Qi's method

rm(list=ls())

args= (commandArgs(TRUE))
if (length(args) == 0) {
  print("No arguments supplied.")
  MODEL = "2_40"
} else {
  for (i in 1:length(args)) {
    eval(parse(text=args[[i]]))
  }
}

DIRmain <- "/Users/yjiang/Dissertation/Simulation/otherStatistics/"
#MODEL <- "2_40"
dataDIR <- paste0(DIRmain, "oriGenoPedtrait_", MODEL ,"/")
ORname <- paste0("logOR_", MODEL, ".RData")

################################################################
## Read in simulated data sets.
################################################################

setwd(DIRmain)
#install.packages("kinship", lib= "./kinship", type= "source", repos= NULL)
#powerGeno <- get(load("powerGeno.RData"))
#alphaGeno <- get(load("alphaGeno.RData"))
#pedstr <- get(load("pedstrTrait.RData"))
seeds <- get(load("seedFile.RData"))
snp <- get(load("SNPInfo_sub.RData"))
logORrisk <- get(load(ORname))
riskVariantList <- snp[which(logORrisk > 0), "rs"]
protectVariantList <- snp[which(logORrisk < 0), "rs"]

NFILES <- 500
for (f in 1:NFILES) {
  SEED <- seeds[f]
  powerGeno <- get(load(paste0(dataDIR,"powerGeno_",f,".RData")))
  pedstr <- get(load(paste0(dataDIR,"pedstrTrait_",f,".RData")))
  NCpowerGeno <- ncol(powerGeno)
  N <- nrow(pedstr)
  row.names(powerGeno) <- 1:N

  riskGeno <- intersect(colnames(powerGeno),riskVariantList)
  numRisk <- length(riskGeno)
```

```
protectGeno <- intersect(colnames(powerGeno),protectVariantList)
numProtect <- length(protectGeno)


#################################################################
## Compute Schaid's kernel and burden statistics.
## pedgene package in R
#################################################################

require("pedgene")
weight1 <- rep(1,NCpowerGeno)
schaidPed <- pedstr
schaidPed$ID <- NULL
colnames(schaidPed) <- c("ped","person","father","mother","sex","trait")
pedPer <- schaidPed[ ,c("ped","person")]
schaidPowerG <- cbind(pedPer,powerGeno)

schaid_powerBT <- pedgene(schaidPed, schaidPowerG, male.dose= 2, checkpeds= FALSE, weights
    = NULL, weights.mb= FALSE, method= "kounen", acc.davies= 1e-9)
pKernelPowerBT <- schaid_powerBT$pgdf$pval.kernel
pBurdenPowerBT <- schaid_powerBT$pgdf$pval.burden

schaid_powerMB <- pedgene(schaidPed, schaidPowerG, male.dose= 2, checkpeds= FALSE, weights
    = NULL, weights.mb= TRUE, method= "kounen", acc.davies= 1e-9)
pKernelPowerMB <- schaid_powerMB$pgdf$pval.kernel
pBurdenPowerMB <- schaid_powerMB$pgdf$pval.burden

schaid_power1 <- pedgene(schaidPed, schaidPowerG, male.dose= 2, checkpeds= FALSE, weights=
     weight1, weights.mb= TRUE, method= "kounen", acc.davies= 1e-9)
pKernelPower1 <- schaid_power1$pgdf$pval.kernel
pBurdenPower1 <- schaid_power1$pgdf$pval.burden

#################################################################
## Adjust ID coding for the following statistics computation.
#################################################################

pedIDstr <- pedstr
IDindex <- with(pedIDstr,pedigree(id= Per, dadid= Father, momid= Mother, sex= Sex, famid=
    Ped))
pedIDstr$fa <- IDindex$findex
pedIDstr$mo <- IDindex$mindex
pedIDstr <- pedIDstr[ ,c("Ped","ID","fa","mo","Sex","trait")]

#################################################################
## Compute famSKAT statistics.
## http://www.hsph.harvard.edu/han-chen/2014/07/31/famskat/
#################################################################

detach("package:pedgene")
detach("package:kinship2")
require("kinship" ,lib.loc= "./kinship")
source("famSKAT_v1.8_04052013.R")
y <- pedIDstr$trait
fskPowerG <- as.matrix(powerGeno)
id <- pedIDstr$ID
kin <- makekinship(pedIDstr$Ped, pedIDstr$ID, pedIDstr$fa, pedIDstr$mo)

fSKATpowerW <- famSKAT(phenotype= y, genotypes= fskPowerG, id= id, fullkins= kin)
pFamSKATpowerW <- fSKATpowerW$pvalue  # by default: Wu weights SKAT #

fSKATpower1 <- famSKAT(phenotype= y, genotypes= fskPowerG, id= id, fullkins= kin,
    sqrtweights= weight1)
pFamSKATpower1 <- fSKATpower1$pvalue

#################################################################
## Compute Qi's statistics.
## http://www.pitt.edu/~qiy17/Softwares.html
#################################################################
```

```
112
113   source("./Qi_FSKAT/glmmPQL.s")
114   source("./Qi_FSKAT/FSKAT.R")
115   qiPedstr <- pedIDstr[ ,c("ID","fa","mo","trait")]
116   qiPedstr[qiPedstr$fa == 0, "fa"] <- NA
117   qiPedstr[qiPedstr$mo == 0, "mo"] <- NA
118   qiPowerG <- as.data.frame(t(powerGeno))
119   IDname <- colnames(qiPowerG)
120   qiPowerG$snp <- row.names(qiPowerG)
121   qiPowerG$gene <- "simuGene"
122   qiPowerG <- qiPowerG[ ,c("gene","snp",IDname)]
123
124   pQiPowerW <- FSKAT(phenotype= y, genotypes= qiPowerG, id= qiPedstr$ID, fa=qiPedstr$fa, mo=
          qiPedstr$mo, family= "binomial", covariates= NULL, weights= NULL, acc= 1e-9)
125   pQiPowerW <- as.numeric(pQiPowerW[ ,2])   # by default: Wu weights SKAT #
126   qiWeight1 <- qiPowerG[ ,c("gene","snp")]
127   qiWeight1$weight <- 1
128   pQiPower1 <- FSKAT(phenotype= y, genotypes= qiPowerG, id= qiPedstr$ID, fa=qiPedstr$fa, mo=
          qiPedstr$mo, family= "binomial", covariates= NULL, weights= qiWeight1, acc= 1e-9)
129   pQiPower1 <- as.numeric(pQiPower1[ ,2])
130
131   detach("package:kinship")
132
133   ################################################################
134   ## Output results.
135   ################################################################
136
137   result <- c(pKernelPowerBT, pBurdenPowerBT, pKernelPowerMB, pBurdenPowerMB, pKernelPower1,
          pBurdenPower1,
138               pFamSKATpowerW, pFamSKATpower1, pQiPowerW, pQiPower1, NCpowerGeno, numRisk,
                      numProtect, SEED)
139   otherStats <- as.data.frame(t(result))
140   outputFileName <- paste0("powerOther_", MODEL, ".txt")
141   write.table(otherStats, file= outputFileName, append= TRUE, quote= FALSE,
142               sep= "\t", row.names= FALSE, col.names= FALSE)
143 }
```

B.4 PLOT TYPE I ERROR RATES AND POWER LEVELS

We used the following codes to collect the simulation results and plot type I error rates and power levels.

```
1  ################################################################
2  ## Draw barplots of power and type I error rates.
3  ################################################################
4
5  rm(list=ls())
6
7  require("binom")
8  require("ggplot2")
9  require("RColorBrewer")
10 require("grid")
11
```

```
Scenario <- c("2.0/0.8\n30%/0%","2.0/0.8\n30%/20%","m10\n30%/0%","m10\n30%/20%","m4\n30%/0%","m4\n30%/20%","m3\n30%/0%","m3\n30%/20%")
Model <- c("2_30","2_30_08_20","m_30","m_30_m_20","m4_30","m4_30_m4_20","m3_30","m3_30_m3_20")
fileName5 <- "power_30_5.pdf"; fileName1 <- "power_30_1.pdf"
fileName5_alpha <- "alpha_30_5.pdf"; fileName1_alpha <- "alpha_30_1.pdf"

#Scenario <- c("2.0/0.8\n40%/0%","2.0/0.8\n40%/20%","m10\n40%/0%","m10\n40%/20%","m4\n40%/0%","m4\n40%/20%","m3\n40%/0%","m3\n40%/20%")
#Model <- c("2_40","2_40_08_20","m_40","m_40_m_20","m4_40","m4_40_m4_20","m3_40","m3_40_m3_20")
#fileName5 <- "power_40_5.pdf"; fileName1 <- "power_40_1.pdf"
#fileName5_alpha <- "alpha_40_5.pdf"; fileName1_alpha <- "alpha_40_1.pdf"

#Model <- c("2_40_new","2_40_08_20_new","m_40_new","m_40_m_20_new","m4_40_new","m4_40_m4_20_new","m3_40_new","m3_40_m3_20_new")
#fileName5 <- "power_40_5_new.pdf"; fileName1 <- "power_40_1_new.pdf"
#fileName5_alpha <- "alpha_40_5_new.pdf"; fileName1_alpha <- "alpha_40_1_new.pdf"

LEVEL1 <- 0.05 ; LEVEL2 <- 0.01 ; ITER <- 500
mainDIR <- "/Volumes/Macintosh HD 2/Dissertation/Programs/Simulation/"
otherDIR <- paste0(mainDIR,"otherStatistics/result/")
typeIDIR <- paste0(mainDIR,"type_I_error/result/")

################################################################
## Analyze power simulation results from other methods.
################################################################

setwd(otherDIR)

powerTable5 <- powerTable1 <- data.frame(stringsAsFactors= FALSE)
for (m in Model) {
  scenario <- Scenario[which(Model == m)]

  resultDIR <- paste0(mainDIR,"result_", m ,"/")
  resultFlm <- data.frame(flmBS= numeric(0), flmFR= numeric(0), stringsAsFactors= FALSE)
  R <- c(1:ITER)
  for (j in R) {
    r <- get(load(paste0(resultDIR,"simu_result_",j,".RData")))
    resultFlm[j, ] <- r
  }

  otherPowerName <- paste0("powerOther_", m, ".txt")
  schaidFlmName <- paste0("powerSchaidFLM_", m, ".txt")

  other <- read.table(otherPowerName, header= FALSE, sep= "\t", stringsAsFactors= FALSE)
  colnames(other) <- c("kernel_BT", "burden_BT", "kernel_MB", "burden_MB", "kernel_UW", "burden_UW",
                       "famSKAT_W", "famSKAT_UW", "FSKAT_W", "FSKAT_UW", "numTotal", "numRisk", "numProtect", "seed")

  flmSchaid <- read.table(schaidFlmName, header= FALSE, sep= "\t", stringsAsFactors= FALSE)
  colnames(flmSchaid) <- c("flmBS_kernel","flmBS_burden","seed")

  otherPower <- merge(flmSchaid, other, by= "seed", all= TRUE, sort= FALSE)
  otherPower$numTotal <- otherPower$numRisk <- otherPower$numProtect <- otherPower$seed <- NULL

  Power <- cbind(resultFlm, otherPower)
  Nrow <- nrow(Power)
  Ncol <- ncol(Power)
  methods <- colnames(Power)

  scenarioTable5 <- data.frame(stringsAsFactors= FALSE)
  scenarioTable1 <- data.frame(stringsAsFactors= FALSE)
  for (i in 1:Ncol) {
    p <- Power[ ,i]
    p5 <- length(p[p < LEVEL1])
```

```
    p1 <- length(p[p < LEVEL2])
    power5 <- p5/Nrow
    power1 <- p1/Nrow
    ci5 <- binom.confint(p5, Nrow, conf.level= 1-LEVEL1, methods= "exact")
    ci1 <- binom.confint(p1, Nrow, conf.level= 1-LEVEL1, methods= "exact")
    lower5 <- ci5$lower ; upper5 <- ci5$upper
    lower1 <- ci1$lower ; upper1 <- ci1$upper
    testResult5 <- c(power5, lower5, upper5)
    testResult1 <- c(power1, lower1, upper1)
    scenarioTable5 <- rbind(scenarioTable5, testResult5)
    scenarioTable1 <- rbind(scenarioTable1, testResult1)
  }
  colnames(scenarioTable5) <- c("power","lower","upper")
  colnames(scenarioTable1) <- c("power","lower","upper")
  scenarioTable5$method <- scenarioTable1$method <- methods
  scenarioTable5$scenario <- scenarioTable1$scenario <- scenario
  powerTable5 <- rbind(powerTable5, scenarioTable5)
  powerTable1 <- rbind(powerTable1, scenarioTable1)
}

powerTable5$flag <- powerTable1$flag <- "okay"
methodMatch <- unique(powerTable5$method)

#################################################################
## Indicate the inflation or deflation of Type I errors.
#################################################################

setwd(typeIDIR)

alphaTable5 <- alphaTable1 <- data.frame(stringsAsFactors= FALSE)
for (n in Model) {
  scenario <- Scenario[which(Model == n)]

  otherName <- paste0("typeIother_logOR_", n, ".txt")
  other <- read.table(otherName, header= FALSE, sep= "\t", stringsAsFactors= FALSE)
  colnames(other) <- c("kernel_BT","burden_BT","kernel_MB","burden_MB","kernel_UW","
      burden_UW",
                       "famSKAT_W","famSKAT_UW","FSKAT_W","FSKAT_UW","seed")
  flmName <- paste0("typeIflm_logOR_", n, ".txt")
  flm <- read.table(flmName, header= FALSE, sep= "\t", stringsAsFactors= FALSE)
  colnames(flm) <- c("flmBS","flmFR","flmBS7","flmFR7","flmBS9","flmFR9","numRisk","
      numProtect","seed")
  flm$flmBS7 <- flm$flmFR7 <- flm$flmBS9 <- flm$flmFR9 <- NULL

  #flm3 <- read.table("typeIflm_3.txt", header= FALSE, sep= "\t", stringsAsFactors= FALSE)
  #colnames(flm3) <- c("BS3alpha","FR3alpha","causal","seed")

  error <- flm
  error <- merge(other, flm, by= "seed", all= TRUE, sort= FALSE)
  #error <- merge(error, flm3, by= "seed", all= TRUE, sort= FALSE)

  flmEmbedName <- paste0("typeIschaidFlm_logOR_", n, ".txt")
  flmEmbed <- read.table(flmEmbedName, header= FALSE, sep= "\t", stringsAsFactors= FALSE)
  colnames(flmEmbed) <- c("flmBS3_kernel","flmBS3_burden","flmBS_kernel","flmBS_burden",
                          "flmBS7_kernel","flmBS7_burden","flmBS9_kernel","flmBS9_burden","
                              seed")
  flmEmbed5 <- flmEmbed[ ,c("flmBS_kernel","flmBS_burden","seed")]
  error <- merge(error, flmEmbed5, by= "seed", all= TRUE, sort= FALSE)

  error$seed <- error$numRisk <- error$numProtect <- NULL
  NCerror <- ncol(error)
  NRerror <- nrow(error)

  ci5data <- ci1data <- data.frame(stringsAsFactors= FALSE)
  for (i in 1:NCerror) {
    nAlpha <- error[ ,i]
    x5 <- length(nAlpha[nAlpha < LEVEL1])
    x1 <- length(nAlpha[nAlpha < LEVEL2])
```

```
    ci5 <- binom.confint(x5, NRerror, conf.level= 1-LEVEL1, methods= "exact")
    ci1 <- binom.confint(x1, NRerror, conf.level= 1-LEVEL1, methods= "exact")
    ci5data <- rbind(ci5data,ci5)
    ci1data <- rbind(ci1data,ci1)
  }
  method <- colnames(error)[1:NCerror]
  ci5data$method <- ci1data$method <- method
  ci5data$flag <- ci1data$flag <- "okay"
  ci5data[ci5data$lower > LEVEL1, "flag"] <- "I"
  ci5data[ci5data$upper < LEVEL1, "flag"] <- "D"
  ci1data[ci1data$lower > LEVEL2, "flag"] <- "I"
  ci1data[ci1data$upper < LEVEL2, "flag"] <- "D"

  stat <- Scenario[which(Model == n)]

  inflate5 <- ci5data[ci5data$flag == "I","method"]
  deflate5 <- ci5data[ci5data$flag == "D","method"]
  powerTable5[(powerTable5$scenario == stat) & (powerTable5$method %in% inflate5), "flag"]
      <- "I"
  powerTable5[(powerTable5$scenario == stat) & (powerTable5$method %in% deflate5), "flag"]
      <- "D"
  inflate1 <- ci1data[ci1data$flag == "I","method"]
  deflate1 <- ci1data[ci1data$flag == "D","method"]
  powerTable1[(powerTable1$scenario == stat) & (powerTable1$method %in% inflate1), "flag"]
      <- "I"
  powerTable1[(powerTable1$scenario == stat) & (powerTable1$method %in% deflate1), "flag"]
      <- "D"

  ci5data <- ci5data[match(methodMatch, ci5data$method), ]
  ci1data <- ci1data[match(methodMatch, ci5data$method), ]

  ci5data$scenario <- scenario ; ci1data$scenario <- scenario
  alphaTable5 <- rbind(alphaTable5, ci5data)
  alphaTable1 <- rbind(alphaTable1, ci1data)

}

powerTable5[powerTable5$flag == "okay", "flag"] <- " "
powerTable1[powerTable1$flag == "okay", "flag"] <- " "

powerTable5$order <- factor(powerTable5$scenario, unique(powerTable5$scenario))
powerTable1$order <- factor(powerTable1$scenario, unique(powerTable1$scenario))

alphaTable5 <- alphaTable5[ ,c("mean", "method", "scenario")]
alphaTable1 <- alphaTable1[ ,c("mean", "method", "scenario")]

alphaTable5$order <- factor(alphaTable5$scenario, unique(alphaTable5$scenario))
alphaTable1$order <- factor(alphaTable1$scenario, unique(alphaTable1$scenario))

bound5 <- binom.confint(LEVEL1*NRerror, NRerror, conf.level= 1-LEVEL1, methods= "exact")
UP5 <- bound5$upper ; LO5 <- bound5$lower
bound1 <- binom.confint(LEVEL2*NRerror, NRerror, conf.level= 1-LEVEL1, methods= "exact")
UP1 <- bound1$upper ; LO1 <- bound1$lower

#################################################################
## 30% or 40% models with alpha = 0.05
#################################################################

setwd(mainDIR)
palette <- colorRampPalette(brewer.pal(6, "Paired"))
limits <- aes(ymin= powerTable5$lower, ymax= powerTable5$upper)
powerPlot <- ggplot(powerTable5, aes(x= factor(method, levels= methods), y= power))
  powerPlot <- powerPlot + ggtitle("Power Estimation by Different Statistics")
  powerPlot <- powerPlot + theme(plot.title= element_text(size= 18, face= "bold", vjust= 2))
  powerPlot <- powerPlot + geom_bar(aes(fill= factor(method, levels= methods), order= factor
      (method, levels= methods)), position= "dodge", stat= "identity")
  powerPlot <- powerPlot + facet_wrap(~order, ncol= 2)
  powerPlot <- powerPlot + xlab("Statistics with alpha = 0.05")
```

```
  powerPlot <- powerPlot + theme(text= element_text(size= 16), axis.title.x= element_text(
      vjust= -2))
  powerPlot <- powerPlot + geom_text(aes(x= method, y= 0.01, label= flag, position= "dodge")
      , size= 4)
  powerPlot <- powerPlot + ylab("Power")
  powerPlot <- powerPlot + ylim(0, 1.0)
  powerPlot <- powerPlot + theme(text= element_text(size= 16), axis.title.y= element_text(
      vjust= 3))
  powerPlot <- powerPlot + theme(axis.ticks= element_blank(), axis.text.x= element_blank(),
      strip.text.x= element_text(size= 16, colour= "darkblue"))
  powerPlot <- powerPlot + geom_errorbar(limits, width= 0.25)
  powerPlot <- powerPlot + guides(fill= guide_legend(title= "Methods", keywidth = 1.62,
      keyheight = 1))
  powerPlot <- powerPlot + guides(colour= guide_legend(title.hjust= 0.5))
  powerPlot <- powerPlot + scale_fill_manual(values= palette(length(methods)))
  powerPlot <- powerPlot + theme(plot.background= element_rect(fill= "lightgrey"),
                                 plot.margin = unit(c(2, 1.5, 2, 1.5), "cm"))
ggsave(filename= fileName5, plot= powerPlot, width= 16, height= 20)

alphaPlot <- ggplot(alphaTable5, aes(x= factor(method, levels= methods), y= mean))
alphaPlot <- alphaPlot + ggtitle("Type I Error Estimation by Different Statistics")
alphaPlot <- alphaPlot + theme(plot.title= element_text(size= 18, face= "bold", vjust= 2))
alphaPlot <- alphaPlot + geom_bar(aes(fill= factor(method, levels= methods), order= factor(
    method, levels= methods)), position= "dodge", stat= "identity")
alphaPlot <- alphaPlot + facet_wrap(~order, ncol= 2)
alphaPlot <- alphaPlot + xlab("Statistics with alpha = 0.05")
alphaPlot <- alphaPlot + theme(text= element_text(size= 16), axis.title.x= element_text(
    vjust= -2))
alphaPlot <- alphaPlot + ylab("Type I error rates")
#alphaPlot <- alphaPlot + ylim(0, max(alphaTable5$mean))
alphaPlot <- alphaPlot + coord_cartesian(ylim= c(0,0.20))
alphaPlot <- alphaPlot + geom_hline(yintercept= c(UP5,LO5), linetype="dashed", colour= "
    brown")
alphaPlot <- alphaPlot + geom_hline(yintercept= 0.05, linetype= "dashed", colour= "black")
alphaPlot <- alphaPlot + theme(text= element_text(size= 16), axis.title.y= element_text(
    vjust= 3))
alphaPlot <- alphaPlot + theme(axis.ticks= element_blank(), axis.text.x= element_blank(),
    strip.text.x= element_text(size= 16, colour= "darkblue"))
alphaPlot <- alphaPlot + guides(fill= guide_legend(title= "Methods", keywidth = 1.62,
    keyheight = 1))
alphaPlot <- alphaPlot + guides(colour= guide_legend(title.hjust= 0.5))
alphaPlot <- alphaPlot + scale_fill_manual(values= palette(length(methods)))
alphaPlot <- alphaPlot + theme(plot.background= element_rect(fill= "lightgrey"),
                               plot.margin = unit(c(2, 1.5, 2, 1.5), "cm"))
ggsave(filename= fileName5_alpha, plot= alphaPlot, width= 16, height= 20)

#################################################################
## 30% or 40% models with alpha = 0.01
#################################################################

#setwd(mainDIR)
#palette <- colorRampPalette(brewer.pal(6, "Paired"))
limits <- aes(ymin= powerTable1$lower, ymax= powerTable1$upper)
powerPlot <- ggplot(powerTable1, aes(x= factor(method, levels= methods), y= power))
  powerPlot <- powerPlot + ggtitle("Power Estimation by Different Statistics")
  powerPlot <- powerPlot + theme(plot.title= element_text(size= 18, face= "bold", vjust= 2))
  powerPlot <- powerPlot + geom_bar(aes(fill= factor(method, levels= methods), order= factor
      (method, levels= methods)), position= "dodge", stat= "identity")
  powerPlot <- powerPlot + facet_wrap(~order, ncol= 2)
  powerPlot <- powerPlot + xlab("Statistics with alpha = 0.01")
  powerPlot <- powerPlot + theme(text= element_text(size= 16), axis.title.x= element_text(
      vjust= -2))
  powerPlot <- powerPlot + geom_text(aes(x= method, y= 0.01, label= flag, position= "dodge")
      , size= 4)
  powerPlot <- powerPlot + ylab("Power")
  powerPlot <- powerPlot + ylim(0, 1.0)
  powerPlot <- powerPlot + theme(text= element_text(size= 16), axis.title.y= element_text(
      vjust= 3))
```

```
  powerPlot <- powerPlot + theme(axis.ticks= element_blank(), axis.text.x= element_blank(),
      strip.text.x= element_text(size= 16, colour= "darkblue"))
  powerPlot <- powerPlot + geom_errorbar(limits, width= 0.25)
  powerPlot <- powerPlot + guides(fill= guide_legend(title= "Methods", keywidth = 1.62,
      keyheight = 1))
  powerPlot <- powerPlot + guides(colour= guide_legend(title.hjust= 0.5))
  powerPlot <- powerPlot + scale_fill_manual(values= palette(length(methods)))
  powerPlot <- powerPlot + theme(plot.background= element_rect(fill= "lightgrey"),
                                 plot.margin = unit(c(2, 1.5, 2, 1.5), "cm"))
ggsave(filename= fileName1, plot= powerPlot, width= 16, height= 20)

alphaPlot <- ggplot(alphaTable1, aes(x= factor(method, levels= methods), y= mean))
alphaPlot <- alphaPlot + ggtitle("Type I Error Estimation by Different Statistics")
alphaPlot <- alphaPlot + theme(plot.title= element_text(size= 18, face= "bold", vjust= 2))
alphaPlot <- alphaPlot + geom_bar(aes(fill= factor(method, levels= methods), order= factor(
    method, levels= methods)), position= "dodge", stat= "identity")
alphaPlot <- alphaPlot + facet_wrap(~order, ncol= 2)
alphaPlot <- alphaPlot + xlab("Statistics with alpha = 0.01")
alphaPlot <- alphaPlot + theme(text= element_text(size= 16), axis.title.x= element_text(
    vjust= -2))
alphaPlot <- alphaPlot + ylab("Type I error rates")
#alphaPlot <- alphaPlot + ylim(0, max(alphaTable1$mean))
alphaPlot <- alphaPlot + coord_cartesian(ylim= c(0,0.05))
alphaPlot <- alphaPlot + geom_hline(yintercept= c(UP1,LO1), linetype="dashed", colour= "
    brown")
alphaPlot <- alphaPlot + geom_hline(yintercept= 0.01, linetype="dashed", colour= "black")
alphaPlot <- alphaPlot + theme(text= element_text(size= 16), axis.title.y= element_text(
    vjust= 3))
alphaPlot <- alphaPlot + theme(axis.ticks= element_blank(), axis.text.x= element_blank(),
    strip.text.x= element_text(size= 16, colour= "darkblue"))
alphaPlot <- alphaPlot + guides(fill= guide_legend(title= "Methods", keywidth = 1.62,
    keyheight = 1))
alphaPlot <- alphaPlot + guides(colour= guide_legend(title.hjust= 0.5))
alphaPlot <- alphaPlot + scale_fill_manual(values= palette(length(methods)))
alphaPlot <- alphaPlot + theme(plot.background= element_rect(fill= "lightgrey"),
                               plot.margin = unit(c(2, 1.5, 2, 1.5), "cm"))
ggsave(filename= fileName1_alpha, plot= alphaPlot, width= 16, height= 20)
```

B.5 REAL DATA ANALYSIS

B.5.1 GFLMM approach

We ran the following codes on the Ubuntu PC operating system (GNU/Linux 3.2.0-51-generic x86_64) to conduct a GWAS by using the GFLMM developed in Section 3.1.1.

```
#### This program would compute p-values for each gene in GARM data

rm(list=ls())

args= (commandArgs(TRUE))
if (length(args) == 0) {
  print("No arguments supplied.")
  geneNum = 1
```

```
} else {
  for (i in 1:length(args)) {
    eval(parse(text=args[[i]]))
  }
}

DIR <- "/mnt/clusterfs/users/yij5/Input"
Rpkg <- "/mnt/clusterfs/users/yij5/Rpkg/"
Uname <- "u_matrix_star.RData"
phenoFileName <- "pheno_data.txt"

setwd(DIR)

require(fda, lib.loc= Rpkg)
require(MASS, lib.loc= Rpkg)
require(Matrix, lib.loc= Rpkg)

## parameters ##
order <- 4; basis <- 5
order3 <- 3 ; basis3 <- 3   # if number of variants <= 5 #

NonmissingPheID <- get(load("NonmissingPheID.RData"))
genoData <- get(load("exomeAMD_cleaned_noMissing.RData"))
refGene <- read.table("refseq_genes.txt", header= TRUE, stringsAsFactors= FALSE)
snpAnnot <- get(load("snpAnnotDataV5.RData"))

snpAnnot <- snpAnnot[snpAnnot$filterReason == "okay", ]

## generate genotypes for a certain gene ##
geno <- genoData[NonmissingPheID, ]
geno$PAT <- geno$MAT <- geno$SEX <- geno$PHENOTYPE <- NULL
geno$FID <- geno$IID <- NULL
colnames(geno) <- sub("_.*","",colnames(geno))

## extract rare variants for analysis ##
rareMarker <- snpAnnot[snpAnnot$MAF < 0.05, "marker"]
rareMarker <- intersect(rareMarker,colnames(geno))
geno <- geno[ ,rareMarker]

refGene$name <- refGene$cdsStart <-refGene$cdsEnd <- NULL
refGene <- refGene[nchar(refGene$chrom) <= 5, ]
refGene <- refGene[refGene$chrom != "chrX" & refGene$chrom != "chrY", ]
refGene <- refGene[!duplicated(refGene), ]

testGene <- refGene[geneNum, ]

## get positions of variants within a gene ##
posAnnot <- snpAnnot[ ,c("marker","rsid_Hum","pos_Hum")]
posAnnot <- posAnnot[posAnnot$pos_Hum >= testGene$txStart &
           posAnnot$pos_Hum <= testGene$txEnd &
           posAnnot$chrom %in% testGene$chrom, ]
posAnnot <- posAnnot[order(posAnnot$pos_Hum, decreasing= FALSE), ]
selectMarker <- intersect(colnames(geno),posAnnot$marker)


results <- data.frame(chr= character(0), gene= character(0), nvariants= numeric(0),
                      start= numeric(0), end= numeric(0), pBspline= numeric(0),
                      pFourier= numeric(0), geneID= numeric(0), stringsAsFactors= FALSE)

if (length(selectMarker) > 1) {
  geno <- geno[ ,selectMarker]

  snpNumber <- ncol(geno)
  for (k in 1:snpNumber) {
    marker <- geno[ ,k]
    genotype0 <- sum(marker == 0, na.rm= TRUE)
    genotype2 <- sum(marker == 2, na.rm= TRUE)
    if (genotype0 < genotype2) {
```

```
      geno[ ,k] <- 2-geno[ ,k]
    }    # IMPORTANT: rare allele coded as 2 #
  }

  ## Apply Fan's beta-smooth only model ##
#  geno[is.na(geno)] <- 0    # code missing genotype to 0 #
  pos <- posAnnot[posAnnot$marker %in% selectMarker, "pos_Hum"]

  ## QR decomposition. We do not conduct this. ##
#  dqr     <- qr(geno)
#  index   <- dqr$pivot[1:dqr$rank]
#  geno    <- geno[ ,index]
#  pos     <- pos[index]
  maf <- colMeans(geno)
  pos <- pos[maf > 0]
  if (length(pos) >= 2) {     # at least 2 non-polymorphic variants #

    geno <- geno[ ,maf > 0]     # remove nonpolymorphic variants #
    nsample <- nrow(geno)
    nsnp    <- ncol(geno)

    ## normalize position ##
    if (max(pos) > 1) {
      pos <- (pos - min(pos)) / (max(pos) - min(pos))
    }

    ## generate basis functions ##
    if (nsnp > basis) {     # number of non-polymorphic variants > 5 #
      betabasis <- create.bspline.basis(norder= order, nbasis= basis)
      fourierbasis <- create.fourier.basis(c(0,1), nbasis= basis)
    } else {
      betabasis <- create.bspline.basis(norder= order3, nbasis= basis3)
      fourierbasis <- create.fourier.basis(c(0,1), nbasis= basis3)
    }
    B <- eval.basis(pos, betabasis)
    FR <- eval.basis(pos, fourierbasis)

    geno <- as.matrix(geno)
    UJ <- geno %*% B
    UJ_FR <- geno %*% FR

    ## make sure UJ has full rank of bbasis or fbasis ##
    #  UJdqr <- qr(UJ)
    #  UJdqr_FR <- qr(UJ_FR)
    #  UJindex <- UJdqr$pivot[1:UJdqr$rank]
    #  UJindex_FR <- UJdqr_FR$pivot[1:UJdqr_FR$rank]
    #  UJ <- UJ[ ,UJindex]
    #  UJ_FR <- UJ_FR[ ,UJindex_FR]

    estSigma <- 1.866244e-01; estBetaNull <- 1.449921e+00; lnL <- -6.224243e+02
    Z <- 1        # covariate vector #
    C <- 350000   # number of cubature sizes #
    W <- 1/C      # equal weights #

    phenoFile <- read.table(phenoFileName, header= FALSE, sep= "\t", stringsAsFactors= FALSE
        )
    colnames(phenoFile) <- c("ID","trait")
    NRpheno <- nrow(phenoFile)    # 1275 #
    y <- phenoFile$trait

    uMat <- get(load(Uname))

    numG    <- ncol(UJ)
    numG_FR <- ncol(UJ_FR)

    G    <- as.data.frame(UJ)
    G_FR <- as.data.frame(UJ_FR)

```

```
144     G$Z <- 1
145     G_FR$Z <- 1
146
147     S_theta_numerator <- S_theta_numerator_FR <- 0
148     I_theta_numerator <- I_theta_numerator_FR <- 0
149
150     S_theta_denorminator <- exp(lnL-log(W))
151     I_theta_denorminator <- S_theta_denorminator
152
153     for (c in 1:C) {
154       ## compute p ##
155       u_vec <- uMat[((c-1)*NRpheno+1):(c*NRpheno)]
156       B <- estBetaNull+estSigma*u_vec
157       p <- exp(B)/(1+exp(B))
158
159       ## compute likelihood given c: exp(l_a_c) ##
160       l_c <- sum(y*log(p)+(1-y)*log(1-p))
161       exp_l_c <- exp(l_c)
162
163       G$u_vec    <- u_vec
164       G_FR$u_vec <- u_vec
165
166       t_z_theta    <- as.matrix(G)
167       t_z_theta_FR <- as.matrix(G_FR)
168
169       ## compute z= (G,1,a(n))' and Dl_theta= sum((y-p)*z) ## #
170       y_p <- matrix((y-p), nrow= 1, ncol= NRpheno, byrow= TRUE)
171       Dl_theta    <- (y_p) %*% t_z_theta
172       Dl_theta_FR <- (y_p) %*% t_z_theta_FR
173
174       ## assume equal weights ##
175       S_theta_numerator    <- S_theta_numerator    + Dl_theta*exp_l_c
176       S_theta_numerator_FR <- S_theta_numerator_FR + Dl_theta_FR*exp_l_c
177
178       ## D2l_theta= sum((p^2-p)*z*z') ##
179       z_theta    <- t(t_z_theta)
180       z_theta_FR <- t(t_z_theta_FR)
181
182       row_z_theta    <- nrow(z_theta)
183       row_z_theta_FR <- nrow(z_theta_FR)
184
185       p2_p <- p^2-p
186       p2_p    <- matrix(rep(p2_p,row_z_theta), nrow= row_z_theta, ncol= NRpheno, byrow= TRUE
              )
187       p2_p_FR <- matrix(rep(p2_p,row_z_theta_FR), nrow= row_z_theta_FR, ncol= NRpheno, byrow
              = TRUE)
188
189       D2l_theta    <- (p2_p*z_theta) %*% t_z_theta
190       D2l_theta_FR <- (p2_p_FR*z_theta_FR) %*% t_z_theta_FR
191
192       I_theta_numerator    <- I_theta_numerator + (D2l_theta + t(Dl_theta) %*% Dl_theta)*
              exp_l_c
193       I_theta_numerator_FR <- I_theta_numerator_FR + (D2l_theta_FR + t(Dl_theta_FR) %*%
              Dl_theta_FR)*exp_l_c
194     }
195
196     ## write a function to compute p-value of the score test ##
197     scoreP <- function(S_theta_numerator, I_theta_numerator, numG) {
198
199       S_theta <- S_theta_numerator / S_theta_denorminator
200       I_theta <- -(I_theta_numerator/I_theta_denorminator - t(S_theta) %*% S_theta)
201       S_gamma <- S_theta[ ,1:numG, drop= FALSE]
202
203       I_gamma_gamma <- I_theta[1:numG,1:numG, drop= FALSE]
204       I_gamma_eta   <- I_theta[1:numG,(numG+1):ncol(I_theta), drop= FALSE]
205       I_eta_gamma   <- I_theta[(numG+1):nrow(I_theta),1:numG, drop= FALSE]
206       I_eta_eta     <- I_theta[(numG+1):nrow(I_theta),(numG+1):ncol(I_theta), drop= FALSE]
207       inv_I_eta_eta <- ginv(I_eta_eta)
```

```
      I_gamma <- I_gamma_gamma- I_gamma_eta %*% inv_I_eta_eta %*% I_eta_gamma
      inv_I_gamma <- ginv(I_gamma)

      R <- S_gamma %*% inv_I_gamma %*% t(S_gamma)
      R <- c(R)
      pValue_R <- pchisq(R, df= numG, lower.tail= FALSE)

      return(pValue_R)
    }

    pBspline <- scoreP(S_theta_numerator,I_theta_numerator,numG)
    pFourier <- scoreP(S_theta_numerator_FR,I_theta_numerator_FR,numG_FR)

    ## read out the results ##
    chr   <- testGene$chrom
    gene  <- testGene$name2
    start <- testGene$txStart
    end   <- testGene$txEnd
    nvariants <- nsnp

    results[1, ] <- c(chr, gene, nvariants, start, end, pBspline, pFourier, geneNum)
    write.table(results, file= "/mnt/clusterfs/users/yij5/Output/okay_results_rare_nbasis_5.
        txt",
                append= TRUE, row.names= FALSE, col.names= FALSE, quote= FALSE)
  }
}
```

B.5.2 Embedded method

We ran the following codes on the Ubuntu PC operating system (GNU/Linux 3.2.0-51-generic x86_64) to conduct a GWAS by using the embedded method developed in Section 3.1.5.

```
#### This program would compute p-values for each gene in GARM data
#### Embed FLM in the retrospective regression, variant-dependent basis

rm(list=ls())

args= (commandArgs(TRUE))
if (length(args) == 0) {
  print("No arguments supplied.")
  geneNum = 2
} else {
  for (i in 1:length(args)) {
    eval(parse(text=args[[i]]))
  }
}

DIR <- "/mnt/clusterfs/users/yij5/Input"
Rpkg <- "/mnt/clusterfs/users/yij5/Rpkg/"
phenoFileName <- "pheno_data.txt"

setwd(DIR)

require("fda", lib.loc= Rpkg)
require("survey",lib.loc= Rpkg)
require("quadprog", lib.loc= Rpkg)
require("CompQuadForm", lib.loc= Rpkg)
require("kinship2", lib.loc= Rpkg)
```

```
require("pedgene", lib.loc= Rpkg)
#require(fda) ;require(pedgene)

## parameters ##
order <- 4 ; Nbasis5 <- 5

NonmissingPheID <- get(load("NonmissingPheID.RData"))
genoData <- get(load("exomeAMD_cleaned_noMissing.RData"))
refGene <- read.table("refseq_genes.txt", header= TRUE, stringsAsFactors= FALSE)
snpAnnot <- get(load("snpAnnotDataV5.RData"))
snpAnnot <- snpAnnot[snpAnnot$filterReason == "okay", ]
#snpAnnot <- snpAnnot[snpAnnot$FILTER == "PASS" & snpAnnot$filterReason == "okay", ]

## generate genotypes for a certain gene ##
geno <- genoData[NonmissingPheID, ]

## generate pedigree structure ##
schaidPed <- geno[ ,c("FID","IID","PAT","MAT","SEX","PHENOTYPE")]
colnames(schaidPed) <- c("ped","person","father","mother","sex","trait")
pedPer <- schaidPed[ ,c("ped","person")]

geno$PAT <- geno$MAT <- geno$SEX <- geno$PHENOTYPE <- NULL
geno$FID <- geno$IID <- NULL
colnames(geno) <- sub("_.*","",colnames(geno))

## extract rare variants for analysis ##
rareMarker <- snpAnnot[snpAnnot$MAF < 0.05, "marker"]
rareMarker <- intersect(rareMarker,colnames(geno))
geno <- geno[ ,rareMarker]

refGene$name <- refGene$cdsStart <-refGene$cdsEnd <- NULL
refGene <- refGene[nchar(refGene$chrom) <= 5, ]
refGene <- refGene[refGene$chrom != "chrX" & refGene$chrom != "chrY", ]
refGene <- refGene[!duplicated(refGene), ]

testGene <- refGene[geneNum, ]

## get positions of variants within a gene ##
posAnnot <- snpAnnot[ ,c("marker","rsid_Hum","pos_Hum","chrom")]
posAnnot <- posAnnot[posAnnot$pos_Hum >= testGene$txStart &
           posAnnot$pos_Hum <= testGene$txEnd &
           posAnnot$chrom %in% testGene$chrom, ]
posAnnot <- posAnnot[order(posAnnot$pos_Hum, decreasing= FALSE), ]
selectMarker <- intersect(colnames(geno),posAnnot$marker)

results <- data.frame(chr= character(0), gene= character(0), nvariants= numeric(0),
                      start= numeric(0), end= numeric(0), pBspline= numeric(0),
                      pFourier= numeric(0), geneID= numeric(0), stringsAsFactors= FALSE)

if (length(selectMarker) > 1) {
  geno <- geno[ ,selectMarker]

  snpNumber <- ncol(geno)
  for (k in 1:snpNumber) {
    marker <- geno[ ,k]
    genotype0 <- sum(marker == 0, na.rm= TRUE)
    genotype2 <- sum(marker == 2, na.rm= TRUE)
    if (genotype0 < genotype2) {
      geno[ ,k] <- 2-geno[ ,k]
    }    # IMPORTANT: rare allele coded as 2 #
  }

  ## Apply Fan's beta-smooth only model ##
  pos <- posAnnot[posAnnot$marker %in% selectMarker, "pos_Hum"]
  maf <- colMeans(geno, na.rm= TRUE)
  pos <- pos[maf > 0]
  if (length(pos) >= 2) {     # at least 2 non-polymorphic variants #

```

```
    geno <- geno[ ,maf > 0]     # remove nonpolymorphic variants #
    nsample <- nrow(geno)
    nsnp    <- ncol(geno)

    ## normalize position ##
    if (max(pos) > 1) {
      pos <- (pos - min(pos)) / (max(pos) - min(pos))
    }

    geno <- as.matrix(geno)
    ## generate basis functions ##
    if (nsnp > order) {   # >= 5 polymorphic variants #
      betabasis <- create.bspline.basis(norder= order, nbasis= Nbasis5)
      B <- eval.basis(pos, betabasis)
      UJ <- geno %*% B
      meanUJ <- colMeans(UJ/2, na.rm= TRUE)
      meanTRUE <- all(meanUJ < 1)
      while (!meanTRUE) {
        Nbasis5 <- Nbasis5 + 1
        betabasis <- create.bspline.basis(norder= order, nbasis= Nbasis5)
        B <- eval.basis(pos, betabasis)
        UJ <- geno %*% B
        meanUJ <- colMeans(UJ/2, na.rm= TRUE)
        meanTRUE <- all(meanUJ < 1)
      }
    } else {
      Nbasis5 <- nsnp
      betabasis <- create.bspline.basis(norder= nsnp, nbasis= Nbasis5)
      B <- eval.basis(pos, betabasis)
      UJ <- geno %*% B
      meanUJ <- colMeans(UJ/2, na.rm= TRUE)
      meanTRUE <- all(meanUJ < 1)
      while (!meanTRUE) {
        Nbasis5 <- Nbasis5 + 1
        betabasis <- create.bspline.basis(norder= nsnp, nbasis= Nbasis5)
        B <- eval.basis(pos, betabasis)
        UJ <- geno %*% B
    meanUJ <- colMeans(UJ/2, na.rm= TRUE)
        meanTRUE <- all(meanUJ < 1)
      }
    }

    weightBS <- rep(1, ncol(UJ))

    schaidG_BS <- cbind(pedPer, UJ)
    schaid_BS_result <- pedgene(schaidPed, schaidG_BS, male.dose= 2, checkpeds= FALSE,
        weights= weightBS, weights.mb= TRUE, method= "kounen", acc.davies=1e-9)
    pKernel_BS <- schaid_BS_result$pgdf$pval.kernel
    pBurden_BS <- schaid_BS_result$pgdf$pval.burden

    ## read out the results ##
    chr   <- testGene$chrom
    gene  <- testGene$name2
    start <- testGene$txStart
    end   <- testGene$txEnd
    nvariants <- nsnp

    results[1, ] <- c(chr, gene, nvariants, start, end, pKernel_BS, pBurden_BS, geneNum)
    write.table(results, file= "/mnt/clusterfs/users/yij5/Output/
        okay_results_rare_flmSchaid_5.txt",
                append= TRUE, row.names= FALSE, col.names= FALSE, quote= FALSE)
  }
}
```

APPENDIX C

C CODES: A MODIFIED FUNCTION FOR THE GLOGS PROGRAM

We modified the "ml_gaussnewton_zero.c" function in the original GLOGS program in terms of adding the step-halving strategy to the Gauss-Newton algorithm, applying a new stopping criterion, and using equal cubature weights to approximate the integral (see details in 3.1.4).

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <mpi.h>
#include <zlib.h>

#include "external_functions.h"

typedef struct {
    double  L;
    double  DL_theta, DL_s;
    double  D2L_theta, D2L_s;
    double  D2L_theta_s;
}   derivative_return;


derivative_return
derivative_binomial_glmm(double theta, double s, int nobs, double *Y,
     int ncube, double *u_vec, double *pu_vec, int *u_index_vec)
{
  double    u, pu;
  int       u_index;
  double    z, p;
  double    L;
  double    Dlz_theta , Dlz_s;
  double    D2lz_theta, D2lz_s;
  double    D2lz_theta_s;
  double    prod_L;
  double    sum_Dlz_theta , sum_Dlz_s;
  double    sum_D2lz_theta, sum_D2lz_s;
  double    sum_D2lz_theta_s;
  double    L_acc;
  double    DL_theta_acc , DL_s_acc;
  double    D2L_theta_acc, D2L_s_acc;
  double    D2L_theta_s_acc;
```

```
36    double    pL_pu;
37              derivative_return ret_struct;
38    int       i, c;
39
40    L_acc = 0;
41    DL_theta_acc = 0;
42    DL_s_acc = 0;
43    D2L_theta_acc = 0;
44    D2L_s_acc = 0;
45    D2L_theta_s_acc = 0;
46
47    for (c = 0; c < ncube; c++) {
48      pu = pu_vec[c];
49      prod_L = 1;
50      sum_Dlz_theta = 0;
51      sum_Dlz_s = 0;
52      sum_D2lz_theta = 0;
53      sum_D2lz_s = 0;
54      sum_D2lz_theta_s = 0;
55      u_index = u_index_vec[c];
56
57      for (i = 0; i < nobs; i++) {
58        u = u_vec[(u_index * nobs) + i];
59        z = theta + s * u;
60        if (exp(z) == INFINITY) {
61          p = 1;
62        } else {
63          p = exp(z) / (1 + exp(z));
64        }
65        L = pow(p, Y[i]) * pow(1-p, 1-Y[i]);
66
67        prod_L = prod_L * L;
68        Dlz_theta = Y[i] - p;
69        Dlz_s = u * Dlz_theta;
70        D2lz_theta = p * p - p;
71        D2lz_theta_s = u * D2lz_theta;
72        D2lz_s = u * D2lz_theta_s;
73
74        sum_Dlz_theta = sum_Dlz_theta + Dlz_theta;
75        sum_Dlz_s = sum_Dlz_s + Dlz_s;
76        sum_D2lz_theta = sum_D2lz_theta + D2lz_theta;
77        sum_D2lz_s = sum_D2lz_s + D2lz_s;
78        sum_D2lz_theta_s = sum_D2lz_theta_s + D2lz_theta_s;
79      }
80
81      pL_pu = prod_L * pu;
82      L_acc = L_acc + pL_pu;
83      DL_theta_acc = DL_theta_acc + sum_Dlz_theta * pL_pu;
84      DL_s_acc = DL_s_acc + sum_Dlz_s * pL_pu;
85      D2L_theta_acc = D2L_theta_acc + (sum_D2lz_theta + sum_Dlz_theta * sum_Dlz_theta) * pL_pu
            ;
86      D2L_s_acc = D2L_s_acc + (sum_D2lz_s + sum_Dlz_s * sum_Dlz_s) * pL_pu;
87      D2L_theta_s_acc = D2L_theta_s_acc + (sum_D2lz_theta_s + sum_Dlz_theta * sum_Dlz_s) *
            pL_pu;
88    }
89
90    ret_struct.L = L_acc;
91    ret_struct.DL_theta = DL_theta_acc;
92    ret_struct.DL_s = DL_s_acc;
93    ret_struct.D2L_theta = D2L_theta_acc;
94    ret_struct.D2L_s = D2L_s_acc;
95    ret_struct.D2L_theta_s = D2L_theta_s_acc;
96
97    return (ret_struct);
98  }
99
100 derivative_return
101 gather_nodes(derivative_return d_ret, int totnodes)
```

```
{
  MPI_Status stat;
  double dtool;
  int    r;

  for (r = 1; r < totnodes; r++) {
    MPI_Recv(&dtool, 1, MPI_DOUBLE, r, r, MPI_COMM_WORLD, &stat);
    d_ret.L = d_ret.L + dtool;

    MPI_Recv(&dtool, 1, MPI_DOUBLE, r, r, MPI_COMM_WORLD, &stat);
    d_ret.DL_theta = d_ret.DL_theta + dtool;
    MPI_Recv(&dtool, 1, MPI_DOUBLE, r, r, MPI_COMM_WORLD, &stat);
    d_ret.DL_s = d_ret.DL_s + dtool;

    MPI_Recv(&dtool, 1, MPI_DOUBLE, r, r, MPI_COMM_WORLD, &stat);
    d_ret.D2L_theta = d_ret.D2L_theta + dtool;
    MPI_Recv(&dtool, 1, MPI_DOUBLE, r, r, MPI_COMM_WORLD, &stat);
    d_ret.D2L_s = d_ret.D2L_s + dtool;

    MPI_Recv(&dtool, 1, MPI_DOUBLE, r, r, MPI_COMM_WORLD, &stat);
    d_ret.D2L_theta_s = d_ret.D2L_theta_s + dtool;
  }

  return (d_ret);
}

void
calculate_derivatives(derivative_return d_ret, double *L, double *DlnL, double *D2lnL)
{
  double    DL_theta, DL_s;
  double    D2L_theta, D2L_s;
  double    D2L_theta_s;
  double    DlnL_theta, DlnL_s;
  double    D2lnL_theta, D2lnL_s;
  double    D2lnL_theta_s;

  *L = d_ret.L;

  DL_theta = d_ret.DL_theta;
  DL_s = d_ret.DL_s;

  D2L_theta = d_ret.D2L_theta;
  D2L_s = d_ret.D2L_s;
  D2L_theta_s = d_ret.D2L_theta_s;

  DlnL_theta = DL_theta / *L;
  DlnL_s = DL_s / *L;

  DlnL[0] = DlnL_theta;
  DlnL[1] = DlnL_s;

  D2lnL_theta = (D2L_theta / *L) - DlnL_theta * DlnL_theta;
  D2lnL_s = (D2L_s / *L) - DlnL_s * DlnL_s;
  D2lnL_theta_s = (D2L_theta_s / *L) - DlnL_theta * DlnL_s;

  D2lnL[0] = -D2lnL_theta;
  D2lnL[3] = -D2lnL_s;
  D2lnL[1] = D2lnL[2] = -D2lnL_theta_s;
}

void
calculate_parameters(double theta, double s, double *DlnL, double *D2lnL,
                     double *theta_star, double *s_star, double alpha)
{
  double  x11, x12, x22;
  double  z1 , z2;
  double  y1 , y2;
  double  denom;
```

```
170
171   x11 = D2lnL[0];
172   x12 = D2lnL[1];
173   x22 = D2lnL[3];
174
175   z1 = DlnL[0];
176   z2 = DlnL[1];
177
178   denom = x11 * x22 - x12 * x12;
179
180   if (denom != 0) {
181     y1 = (x22 * z1 - z2 * x12) / denom;
182     y2 = (-x12 * z1 + x11 * z2) / denom;
183     *theta_star = theta + alpha * y1;
184     *s_star = s + alpha * y2;
185   } else {
186     *theta_star = theta;
187     *s_star = s;
188   }
189 }
190
191 /* Equal weights, no updates */
192 int
193 update_pu(double theta, double s, int nobs,  double *Y, int ncube,
194           double *u_vec, double *pu_vec, int *u_index_vec, double L,
195           int node, int totnodes, double thresh)
196 {
197   double    u, pu;
198   int       u_index;
199   double    z, p;
200   double    L_u, L_prod, pu_0;
201   int       nonzero_cube;
202   double    pu_acc;
203   int       i, c, r, inttool;
204   double    dtool;
205   int       zero_index, nonzero_index;
206   MPI_Status  stat;
207
208   nonzero_cube = 0;
209   for (c = 0; c < ncube; c++) {
210     pu = pu_vec[c];
211     u_index = u_index_vec[c];
212     L_prod = 1;
213
214     for (i = 0; i < nobs; i++) {
215       u = u_vec[(u_index * nobs) + i];
216       z = theta + s * u;
217       if (exp(z) == INFINITY) {
218         p = 1;
219       } else {
220         p = exp(z) / (1 + exp(z));
221       }
222         L_u = pow(p, Y[i]) * pow(1 - p, 1 - Y[i]);
223       L_prod = L_prod * L_u;
224     }
225     pu_0 = (L_prod * pu) / L;
226
227     if ((pu_0 < 0) || (pu_0 > 1)) {
228       printf("Warning: pu_0= %le\n", pu_0);
229       printf("Diagnostic: L_prod= %le pu= %le L_prod*pu= %le L= %le\n", L_prod, pu, L_prod *
              pu, L);
230     }
231     if (pu_0 > 0) {
232       nonzero_cube++;
233     }
234   }
235
236   if (node == 0) {
```

```
237     for (r = 1; r < totnodes; r++) {
238       MPI_Recv(&inttool, 1, MPI_INT, r, r, MPI_COMM_WORLD, &stat);
239       nonzero_cube = nonzero_cube + inttool;
240     }
241     thresh = thresh / (double)(nonzero_cube);
242     for (r = 1; r < totnodes; r++) {
243       MPI_Send(&thresh, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
244     }
245   } else {
246     MPI_Send(&nonzero_cube, 1, MPI_INT, 0, node, MPI_COMM_WORLD);
247     MPI_Recv(&thresh, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
248   }
249
250   pu_acc = 0;
251   nonzero_cube = 0;
252   for (c = 0; c < ncube; c++) {
253     if (pu_vec[c] < thresh) {
254     } else {
255       pu_acc = pu_acc + pu_vec[c];
256       nonzero_cube++;
257     }
258   }
259
260   zero_index = 0;
261   nonzero_index = 0;
262   do {
263     while ((zero_index < ncube) && (pu_vec[zero_index] != 0)) {
264       zero_index++;
265     }
266     if (zero_index < ncube) {
267       if (nonzero_index < zero_index)
268         nonzero_index = zero_index;
269       while ((nonzero_index < ncube) && (pu_vec[nonzero_index] == 0)) {
270         nonzero_index++;
271       }
272       if (nonzero_index < ncube) {
273         inttool = u_index_vec[zero_index];
274         u_index_vec[zero_index] = u_index_vec[nonzero_index];
275         u_index_vec[nonzero_index] = inttool;
276       } else {
277         zero_index = nonzero_index;
278       }
279     }
280   } while (zero_index < ncube);
281
282   if (node == 0) {
283     for (r = 1; r < totnodes; r++) {
284       MPI_Recv(&dtool, 1, MPI_DOUBLE, r, r, MPI_COMM_WORLD, &stat);
285       pu_acc = pu_acc + dtool;
286     }
287     for (r = 1; r < totnodes; r++) {
288       MPI_Send(&pu_acc, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
289     }
290   } else {
291     MPI_Send(&pu_acc, 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
292     MPI_Recv(&pu_acc, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
293   }
294
295   return (nonzero_cube);
296 }
297
298 void
299 write_pu(int ncube, int nobs, double *pu_vec, double *u_vec, int *u_index_vec,
300          char *u_outputfile, char *pu_outputfile, char *write_append)
301 {
302   int   c, r, u_index;
303         FILE *fptr_pu;
304         gzFile fptr_u ;
```

```
}

int
main(int argc, char *argv[])
{
  int       node, totnodes;
  int       nobs, ngene, ncube_node_base;
  double    *phenotypes;
  double    *u, pu_base, *pu;
  int       *u_index_vec;
  double    theta_base, s_lo, s_hi, s_step, s_curr;
  double    cull_thresh;
  int       *id, *u_id, haltcrit;
  int       ncube_node;
  double    theta, s;
            derivative_return d_ret;
  double    L, DlnL_distance, DlnL[2], D2lnL[4];
  double    theta_0, s_0;
            derivative_return d_ret_0;
  double    L_0, DlnL_distance_0, DlnL_0[2], D2lnL_0[4];
  double    difference;
  double    theta_store, s_store;
  double    L_store, DlnL_store[2], D2lnL_store[4];
  int       ncube_node_store;
  double    *pu_store;
  int       *u_index_vec_store;
  double    s_eval_code, stop_code, update_pu_code, pu_store_code, pu_write_code;
  int       retcode;
  double    D2lnLinv[4];
  int       i, j, r;
  char      c, fname[256], fname2[256];
            FILE *fptr;
  double    dtool;
            MPI_Status stat;

  double    lnL_old;
  double    lnL_new;
  double    L_distance2;
  double    L_compare;
  double    FTOL;
  double    EPS;

  double    alpha;
  double    stop_step ;

  EPS = pow(10, -10);

  MPI_Init(&argc, &argv);
  MPI_Comm_size(MPI_COMM_WORLD, &totnodes);
  MPI_Comm_rank(MPI_COMM_WORLD, &node);

  if (argc != 10) {
    if (node == 0) {
      printf("Required arguments: <pheno/cov file> <nobs> <ncube_node> <theta effect>
      <s effect lo> <s effect hi> <s effect step> <stopping crit> <point culling threshhold
          >\n");
    }
    MPI_Finalize();
    exit(0);
  }
  sscanf(argv[2], "%i", &nobs);
  id = (int *)malloc(nobs * sizeof(int));
  phenotypes = (double *)malloc(nobs * sizeof(double));
  fptr = fopen(argv[1], "r");
  for (i = 0; i < nobs; i++) {
    fscanf(fptr, "%i %lf", &(id[i]), &(phenotypes[i]));
  }
  fclose(fptr);
```

```
372
373   sscanf(argv[3], "%i", &ncube_node_base);
374   u_id = (int *)malloc(nobs * sizeof(int));
375   u = (double *)malloc(ncube_node_base * nobs * sizeof(double));
376   sprintf(fname, "u_matrix_%i.dat", node);
377   fptr = fopen(fname, "r");
378   for (i = 0; i < nobs; i++) {
379     if (fscanf(fptr, "%i", &(u_id[i])) == EOF) {
380       if (node == 0) {
381         printf("Too small cubature file. Quitting.\n");
382       }
383       MPI_Finalize();
384       exit(0);
385     }
386   }
387   for (i = 0; i < ncube_node_base * nobs; i++) {
388     if (fscanf(fptr, "%lf", &(u[i])) == EOF) {
389       if (node == 0) {
390         printf("Too small cubature file. Quitting.\n");
391       }
392       MPI_Finalize();
393       exit(0);
394     }
395   }
396   fclose(fptr);
397
398   u_index_vec = (int *)malloc(ncube_node_base * sizeof(int));
399   for (i = 0; i < ncube_node_base; i++) {
400     u_index_vec[i] = i;
401   }
402   u_index_vec_store = (int *)malloc(ncube_node_base * sizeof(int));
403   pu = (double *)malloc(ncube_node_base * sizeof(double));
404   pu_store = (double *)malloc(ncube_node_base * sizeof(double));
405   pu_base = 1.0 / (double)(ncube_node_base * totnodes);
406
407   haltcrit = 0;
408   for (i = 0; i < nobs; i++) {
409     if (id[i] != u_id[i])
410       haltcrit = 1;
411   }
412   if (haltcrit == 1) {
413     if (node == 0) {
414       printf("Nonmatching phenotype/covariate and cubature person IDs. Quitting.\n");
415     }
416     MPI_Finalize();
417     exit(0);
418   }
419   sscanf(argv[4], "%le", &theta_base);
420   sscanf(argv[5], "%le", &s_lo);
421   sscanf(argv[6], "%le", &s_hi);
422   sscanf(argv[7], "%le", &s_step);
423   sscanf(argv[8], "%le", &FTOL);
424   sscanf(argv[9], "%le", &cull_thresh);
425
426   if (node == 0) {
427     s_eval_code = 1;
428     s_curr = s_lo;
429     while (s_eval_code == 1) {
430       for (r = 1; r < totnodes; r++) {
431         MPI_Send(&s_eval_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
432       }
433       printf("s= %le evaluation.\n", s_curr);
434       theta = theta_base;
435       s = s_curr;
436       for (i = 0; i < ncube_node_base; i++) {
437         pu[i] = pu_base;
438       }
439       ncube_node = ncube_node_base;
```

```
440       stop_code = 0;
441       do {
442         for (r = 1; r < totnodes; r++) {
443           MPI_Send(&stop_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
444         }
445         for (r = 1; r < totnodes; r++) {
446           MPI_Send(&theta, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
447           MPI_Send(&s, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
448         }
449         d_ret = derivative_binomial_glmm(theta, s,
450                  nobs, phenotypes, ncube_node, u, pu, u_index_vec);
451         d_ret = gather_nodes(d_ret, totnodes);
452         calculate_derivatives(d_ret, &L, DlnL, D2lnL);
453         DlnL_distance = sqrt(pow(DlnL[0], 2) + pow(DlnL[1], 2));
454         printf("\tlogL = %le, theta = %le, s = %le\n", log(L), theta, s) ;
455
456         stop_step = 0 ;
457         alpha = 1 ;
458         do {
459             for (r = 1; r < totnodes; r++) {
460                 MPI_Send(&stop_step, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
461             }
462             calculate_parameters(theta, s, DlnL, D2lnL, &theta_0, &s_0, alpha);
463             for (r = 1; r < totnodes; r++) {
464                 MPI_Send(&theta_0, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
465                 MPI_Send(&s_0, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
466             }
467             d_ret_0 = derivative_binomial_glmm(theta_0, s_0, nobs, phenotypes,
468                                                ncube_node, u, pu, u_index_vec);
469             d_ret_0 = gather_nodes(d_ret_0, totnodes);
470             calculate_derivatives(d_ret_0, &L_0, DlnL_0, D2lnL_0);
471
472             /* Gauss-Newton with step-halving */
473             if ( log(L_0) - log(L) < -1e-8 )  {
474                 stop_step = 0 ;
475                 alpha = alpha / 2;
476             }
477             else {
478                 stop_step = 1 ;
479             }
480             printf("\talpha = %le, logL_0 = %le, theta = %le, s = %le\n", alpha, log(L_0),
                      theta_0, s_0) ;
481         } while ( stop_step == 0 ) ;
482
483         stop_step = 1 ;
484         for (r = 1; r < totnodes; r++) {
485             MPI_Send(&stop_step, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
486         }
487
488         /* New stopping criteria: Likelihodd comparison */
489         DlnL_distance_0 = sqrt(pow(DlnL_0[0], 2) + pow(DlnL_0[1], 2)) ;
490         difference = DlnL_distance - DlnL_distance_0 ;
491         lnL_new = log(L_0) ;
492         lnL_old = log(L) ;
493         L_distance2 = 2 * fabs(lnL_new - lnL_old) ;
494         L_compare = FTOL * (fabs(lnL_new) + fabs(lnL_old) + EPS) ;
495         /* control FTOL by the command line */
496
497         if (L_distance2 > L_compare) {
498           theta = theta_0;
499           s = s_0;
500
501           L = L_0;
502
503           DlnL_distance = DlnL_distance_0;
504
505           DlnL[0] = DlnL_0[0];
506           DlnL[1] = DlnL_0[1];
```

```
507
508           D2lnL[0] = D2lnL_0[0];
509           D2lnL[1] = D2lnL_0[1];
510           D2lnL[2] = D2lnL_0[2];
511           D2lnL[3] = D2lnL_0[3];
512
513           update_pu_code = 1;
514           for (r = 1; r < totnodes; r++) {
515             MPI_Send(&update_pu_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
516
517             MPI_Send(&theta_0, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
518             MPI_Send(&s_0, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
519             MPI_Send(&L_0, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
520           }
521           ncube_node = update_pu(theta_0, s_0,
522                  nobs, phenotypes,
523           ncube_node, u, pu, u_index_vec, L_0,
524                  node, totnodes, cull_thresh);
525
526           stop_code = 0;
527         } else {
528           update_pu_code = 0;
529           for (r = 1; r < totnodes; r++) {
530             MPI_Send(&update_pu_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
531           }
532
533           if (L_0 > L) {
534               theta = theta_0 ;
535               s = s_0 ;
536               L = L_0 ;
537
538               DlnL_distance = DlnL_distance_0;
539
540               DlnL[0] = DlnL_0[0] ;
541               DlnL[1] = DlnL_0[1] ;
542
543               D2lnL[0] = D2lnL_0[0] ;
544               D2lnL[1] = D2lnL_0[1] ;
545               D2lnL[2] = D2lnL_0[2] ;
546               D2lnL[3] = D2lnL_0[3] ;
547           }
548
549           stop_code = 1;
550         }
551       } while (stop_code == 0);
552
553       for (r = 1; r < totnodes; r++) {
554         MPI_Send(&stop_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
555       }
556
557       if ((s_curr == s_lo) || (L > L_store)) {
558         pu_store_code = 1;
559         for (r = 1; r < totnodes; r++) {
560           MPI_Send(&pu_store_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
561         }
562
563         theta_store = theta;
564         s_store = s;
565
566         L_store = L;
567
568         DlnL_store[0] = DlnL[0];
569         DlnL_store[1] = DlnL[1];
570
571         D2lnL_store[0] = D2lnL[0];
572         D2lnL_store[1] = D2lnL[1];
573         D2lnL_store[2] = D2lnL[2];
574         D2lnL_store[3] = D2lnL[3];
```

```
575
576         ncube_node_store = ncube_node;
577         for (i = 0; i < ncube_node; i++) {
578           pu_store[i] = pu[i];
579           u_index_vec_store[i] = u_index_vec[i];
580         }
581       } else {
582         pu_store_code = 0;
583         for (r = 1; r < totnodes; r++) {
584           MPI_Send(&pu_store_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
585         }
586       }
587
588       printf("Curr Parameters: ");
589       printf("theta= %le s= %le ", theta, s);
590       printf("lnL= %le\n", log(L));
591       printf("Best Parameters: ");
592       printf("theta= %le s= %le ", theta_store, s_store);
593       printf("lnL= %le\n", log(L_store));
594
595       s_curr = s_curr + s_step;
596       if (s_curr > s_hi)
597         s_eval_code = 0;
598     }
599     for (r = 1; r < totnodes; r++) {
600       MPI_Send(&s_eval_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
601     }
602
603     printf("Final Parameters: ");
604     printf("theta= %le s= %le\n", theta_store, s_store);
605     printf("lnL= %le\n", log(L_store));
606     printf("DlnL= %le %le\n", DlnL_store[0], DlnL_store[1]);
607     printf("D2lnL=\n");
608     printf("%le %le\n", D2lnL_store[0], D2lnL_store[1]);
609     printf("%le %le\n", D2lnL_store[2], D2lnL_store[3]);
610     retcode = invert_matrix_2x2(D2lnL_store, D2lnLinv);
611     printf("D2lnLinv=\n");
612     printf("%le %le\n", D2lnLinv[0], D2lnLinv[1]);
613     printf("%le %le\n", D2lnLinv[2], D2lnLinv[3]);
614
615     write_pu(ncube_node_store, nobs, pu_store, u, u_index_vec_store,
616           "u_matrix_star.dat", "pu_matrix_star.dat", "w");
617     for (r = 1; r < totnodes; r++) {
618       MPI_Send(&pu_write_code, 1, MPI_DOUBLE, r, 0, MPI_COMM_WORLD);
619       MPI_Recv(&pu_write_code, 1, MPI_DOUBLE, r, r, MPI_COMM_WORLD, &stat);
620     }
621   } else {
622     MPI_Recv(&s_eval_code, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
623
624     while (s_eval_code == 1) {
625       for (i = 0; i < ncube_node_base; i++) {
626         pu[i] = pu_base;
627       }
628       ncube_node = ncube_node_base;
629
630       MPI_Recv(&stop_code, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
631       while (stop_code == 0) {
632         MPI_Recv(&theta, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
633         MPI_Recv(&s, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
634         d_ret = derivative_binomial_glmm(theta, s, nobs, phenotypes, ncube_node, u, pu,
                u_index_vec);
635         MPI_Send(&(d_ret.L), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
636         MPI_Send(&(d_ret.DL_theta), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
637         MPI_Send(&(d_ret.DL_s), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
638         MPI_Send(&(d_ret.D2L_theta), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
639         MPI_Send(&(d_ret.D2L_s), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
640         MPI_Send(&(d_ret.D2L_theta_s), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
641
```

```
642         MPI_Recv(&stop_step, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
643         while (stop_step == 0) {
644             MPI_Recv(&theta_0, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
645             MPI_Recv(&s_0, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
646             d_ret_0 = derivative_binomial_glmm(theta_0, s_0, nobs, phenotypes, ncube_node,
647                 u, pu, u_index_vec);
648             MPI_Send(&(d_ret_0.L), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
649             MPI_Send(&(d_ret_0.DL_theta), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
650             MPI_Send(&(d_ret_0.DL_s), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
651             MPI_Send(&(d_ret_0.D2L_theta), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
652             MPI_Send(&(d_ret_0.D2L_s), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
653             MPI_Send(&(d_ret_0.D2L_theta_s), 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
654 
655             MPI_Recv(&stop_step, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
656         }
657 
658         MPI_Recv(&update_pu_code, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
659         if (update_pu_code == 1) {
660           MPI_Recv(&theta, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
661           MPI_Recv(&s, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
662           MPI_Recv(&L, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
663 
664           ncube_node = update_pu(theta, s, nobs, phenotypes, ncube_node,
665                                  u, pu, u_index_vec, L, node, totnodes, cull_thresh);
666         }
667         MPI_Recv(&stop_code, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
668       }
669 
670       MPI_Recv(&pu_store_code, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
671       if (pu_store_code == 1) {
672         ncube_node_store = ncube_node;
673         for (i = 0; i < ncube_node; i++) {
674           pu_store[i] = pu[i];
675           u_index_vec_store[i] = u_index_vec[i];
676         }
677       }
678       MPI_Recv(&s_eval_code, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
679     }
680 
681     MPI_Recv(&pu_write_code, 1, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD, &stat);
682     write_pu(ncube_node_store, nobs, pu_store, u, u_index_vec_store,
683              "u_matrix_star.dat", "pu_matrix_star.dat", "a");
684     MPI_Send(&pu_write_code, 1, MPI_DOUBLE, 0, node, MPI_COMM_WORLD);
685   }
686 
687   MPI_Finalize();
688 }
```

BIBLIOGRAPHY

[Abecasis et al., 2002] Abecasis, G. R., Cherny, S. S., Cookson, W. O., and Cardon, L. R. (2002). Merlin–rapid analysis of dense genetic maps using sparse gene flow trees. *Nat. Genet.*, 30(1):97–101.

[Alkan et al., 2011] Alkan, C., Coe, B. P., and Eichler, E. E. (2011). Genome structural variation discovery and genotyping. *Nat. Rev. Genet.*, 12(5):363–376.

[Ambati et al., 2003] Ambati, J., Ambati, B. K., Yoo, S. H., Ianchulev, S., and Adamis, A. P. (2003). Age-related macular degeneration: etiology, pathogenesis, and therapeutic strategies. *Surv Ophthalmol*, 48(3):257–293.

[Ambati et al., 2013] Ambati, J., Atkinson, J. P., and Gelfand, B. D. (2013). Immunology of age-related macular degeneration. *Nat. Rev. Immunol.*, 13(6):438–451.

[Ambati and Fowler, 2012] Ambati, J. and Fowler, B. J. (2012). Mechanisms of age-related macular degeneration. *Neuron*, 75(1):26–39.

[Asimit and Zeggini, 2010] Asimit, J. and Zeggini, E. (2010). Rare variant association analysis methods for complex traits. *Annu. Rev. Genet.*, 44:293–308.

[Baas et al., 2010] Baas, D. C., Ho, L., Ennis, S., Merriam, J. E., Tanck, M. W., Uitterlinden, A. G., de Jong, P. T., Cree, A. J., Griffiths, H. L., Rivadeneira, F., Hofman, A., van Duijn, C., Smith, R. T., Barile, G. R., Gorgels, T. G., Vingerling, J. R., Klaver, C. C., Lotery, A. J., Allikmets, R., and Bergen, A. A. (2010). The complement component 5 gene and age-related macular degeneration. *Ophthalmology*, 117(3):500–511.

[Baron et al., 2014] Baron, R. V., Kollar, C., Mukhopadhyay, N., and Weeks, D. E. (2014). Mega2: validated data-reformatting for linkage and association analyses. *Source Code Biol Med*, 9(1):26.

[Basu and Pan, 2011] Basu, S. and Pan, W. (2011). Comparison of statistical tests for disease association with rare variants. *Genet. Epidemiol.*, 35(7):606–619.

[Bevitt et al., 2003] Bevitt, D. J., Mohamed, J., Catterall, J. B., Li, Z., Arris, C. E., Hiscott, P., Sheridan, C., Langton, K. P., Barker, M. D., Clarke, M. P., and McKie, N. (2003). Expression of ADAMTS metalloproteinases in the retinal pigment epithelium derived cell

line ARPE-19: transcriptional regulation by TNFalpha. *Biochim. Biophys. Acta*, 1626(1-3):83–91.

[Black and Clark, 2015] Black, J. R. and Clark, S. J. (2015). Age-related macular degeneration: genome-wide association studies to translation. *Genet. Med.*

[Cascella et al., 2014] Cascella, R., Ragazzo, M., Strafella, C., Missiroli, F., Borgiani, P., Angelucci, F., Marsella, L. T., Cusumano, A., Novelli, G., Ricci, F., and Giardina, E. (2014). Age-related macular degeneration: insights into inflammatory genes. *J Ophthalmol*, 2014:582842.

[Cavalli-Sforza and Bodmer, 1999] Cavalli-Sforza, L. L. and Bodmer, W. F. (1999). *The genetics of human populations*. Courier Corporation.

[Chen et al., 2013] Chen, H., Meigs, J. B., and Dupuis, J. (2013). Sequence kernel association test for quantitative traits in family samples. *Genet. Epidemiol.*, 37(2):196–204.

[Chen et al., 2010] Chen, W., Stambolian, D., Edwards, A. O., Branham, K. E., Othman, M., Jakobsdottir, J., Tosakulwong, N., Pericak-Vance, M. A., Campochiaro, P. A., Klein, M. L., Tan, P. L., Conley, Y. P., Kanda, A., Kopplin, L., Li, Y., Augustaitis, K. J., Karoukis, A. J., Scott, W. K., Agarwal, A., Kovach, J. L., Schwartz, S. G., Postel, E. A., Brooks, M., Baratz, K. H., Brown, W. L., Brucker, A. J., Orlin, A., Brown, G., Ho, A., Regillo, C., Donoso, L., Tian, L., Kaderli, B., Hadley, D., Hagstrom, S. A., Peachey, N. S., Klein, R., Klein, B. E., Gotoh, N., Yamashiro, K., Ferris Iii, F., Fagerness, J. A., Reynolds, R., Farrer, L. A., Kim, I. K., Miller, J. W., Corton, M., Carracedo, A., Sanchez-Salorio, M., Pugh, E. W., Doheny, K. F., Brion, M., Deangelis, M. M., Weeks, D. E., Zack, D. J., Chew, E. Y., Heckenlively, J. R., Yoshimura, N., Iyengar, S. K., Francis, P. J., Katsanis, N., Seddon, J. M., Haines, J. L., Gorin, M. B., Abecasis, G. R., Swaroop, A., Johnson, R. N., Ai, E., McDonald, H. R., Stolarczuk, M., Pavan, P. R., Billiris, K. K., Iyer, M., Menosky, M. M., Pautler, S. E., Millard, S. M., Hubbard, B., Aaberg, T., DuBois, L., Lyon, A., Anderson-Nelson, S., Jampol, L. M., Weinberg, D. V., Munana, A., Rozenbajgier, Z., Orth, D., Cohen, J., MacCumber, M., Figliulo, C., Porcz, L., Folk, J., Boldt, H. C., Russell, S. R., Ivins, R., Hinz, C. J., Barr, C. C., Bloom, S., Jaegers, K., Kritchman, B., Whittington, G., Heier, J., Frederick, A. R., Morley, M. G., Topping, T., Davis, H. L., Bressler, S. B., Bressler, N. M., Doll, W., Trese, M., Capone, A., Garretson, B. R., Hassan, T. S., Ruby, A. J., Ostentoski, T., McCannel, C. A., Ruszcyzyk, M. J., Grand, G., Blinder, K., Holekamp, N. M., Joseph, D. P., Shah, G., Nobel, G. S., Antoszyk, A. N., Browning, D. J., Stallings, A. H., Singerman, L. J., Miller, D., Novak, M., Pendergast, S., Zegarra, H., Schura, S. A., Smith-Brewer, S., Davidorf, F. H., Chambers, R., Chorich, L., Salerno, J., Dreyer, R. F., Ma, C., Kopfer, M. R., Klein, M. L., Wilson, D. J., Nolte, S. K., Grunwald, J. E., Brucker, A. J., Dunaief, J., Fine, S. L., Maguire, A. M., Stoltz, R. A., McRay, M. N., Fish, G. E., Anand, R., Spencer, R., Arnwine, J., Chandra, S. R., Altaweel, M., Blodi, B., Gottlieb, J., Ip, M., Nork, T. M., Perry-Ramond, J., Fine, S. L., Maguire, M. G., Brightwell-Arnold, M., Harkins, S., Peskin, E., Ying, G. S., and Kurinij, N. (2010). Genetic variants near TIMP3 and high-density lipoprotein-associated loci influence susceptibility to age-related macular degeneration. *Proc. Natl. Acad. Sci. U.S.A.*, 107(16):7401–7406.

[Churchill and Doerge, 1994] Churchill, G. A. and Doerge, R. W. (1994). Empirical threshold values for quantitative trait mapping. *Genetics*, 138(3):963–971.

[Cirulli and Goldstein, 2010] Cirulli, E. T. and Goldstein, D. B. (2010). Uncovering the roles of rare variants in common disease through whole-genome sequencing. *Nat. Rev. Genet.*, 11(6):415–425.

[Clerget-Darpoux et al., 1986] Clerget-Darpoux, F., Bonaiti-Pellie, C., and Hochez, J. (1986). Effects of misspecifying genetic parameters in lod score analysis. *Biometrics*, 42(2):393–399.

[Conneally et al., 1985] Conneally, P. M., Edwards, J. H., Kidd, K. K., Lalouel, J. M., Morton, N. E., Ott, J., and White, R. (1985). Report of the Committee on Methods of Linkage Analysis and Reporting. *Cytogenet. Cell Genet.*, 40(1-4):356–359.

[Cordell and Clayton, 2005] Cordell, H. J. and Clayton, D. G. (2005). Genetic association studies. *Lancet*, 366(9491):1121–1131.

[Dawn Teare and Barrett, 2005] Dawn Teare, M. and Barrett, J. H. (2005). Genetic linkage studies. *Lancet*, 366(9490):1036–1044.

[Drichel et al., 2014] Drichel, D., Herold, C., Lacour, A., Ramirez, A., Jessen, F., Maier, W., Noethen, M. M., Leber, M., Vaitsiakhovich, T., and Becker, T. (2014). Rare variant testing of imputed data: an analysis pipeline typified. *Hum. Hered.*, 78(3-4):164–178.

[Edwards, 1992] Edwards, A. (1992). *Likelihood*. Johns Hopkins University Press.

[Edwards et al., 2005] Edwards, A. O., Ritter, R., Abel, K. J., Manning, A., Panhuysen, C., and Farrer, L. A. (2005). Complement factor H polymorphism and age-related macular degeneration. *Science*, 308(5720):421–424.

[Elston and Stewart, 1971] Elston, R. C. and Stewart, J. (1971). A general model for the genetic analysis of pedigree data. *Hum. Hered.*, 21(6):523–542.

[Fan et al., 2014] Fan, R., Wang, Y., Mills, J. L., Carter, T. C., Lobach, I., Wilson, A. F., Bailey-Wilson, J. E., Weeks, D. E., and Xiong, M. (2014). Generalized functional linear models for gene-based case-control association studies. *Genet. Epidemiol.*, 38(7):622–637.

[Fan et al., 2013] Fan, R., Wang, Y., Mills, J. L., Wilson, A. F., Bailey-Wilson, J. E., and Xiong, M. (2013). Functional linear models for association analysis of quantitative traits. *Genet. Epidemiol.*, 37(7):726–742.

[Friedman et al., 2004] Friedman, D. S., O'Colmain, B. J., Munoz, B., Tomany, S. C., McCarty, C., de Jong, P. T., Nemesure, B., Mitchell, P., and Kempen, J. (2004). Prevalence of age-related macular degeneration in the United States. *Arch. Ophthalmol.*, 122(4):564–572.

[Fritsche et al., 2013] Fritsche, L. G., Chen, W., Schu, M., Yaspan, B. L., Yu, Y., Thorleifsson, G., Zack, D. J., Arakawa, S., Cipriani, V., Ripke, S., Igo, R. P., Buitendijk, G. H., Sim, X., Weeks, D. E., Guymer, R. H., Merriam, J. E., Francis, P. J., Hannum, G., Agarwal, A., Armbrecht, A. M., Audo, I., Aung, T., Barile, G. R., Benchaboune, M., Bird, A. C., Bishop, P. N., Branham, K. E., Brooks, M., Brucker, A. J., Cade, W. H., Cain, M. S., Campochiaro, P. A., Chan, C. C., Cheng, C. Y., Chew, E. Y., Chin, K. A., Chowers, I., Clayton, D. G., Cojocaru, R., Conley, Y. P., Cornes, B. K., Daly, M. J., Dhillon, B., Edwards, A. O., Evangelou, E., Fagerness, J., Ferreyra, H. A., Friedman, J. S., Geirsdottir, A., George, R. J., Gieger, C., Gupta, N., Hagstrom, S. A., Harding, S. P., Haritoglou, C., Heckenlively, J. R., Holz, F. G., Hughes, G., Ioannidis, J. P., Ishibashi, T., Joseph, P., Jun, G., Kamatani, Y., Katsanis, N., N Keilhauer, C., Khan, J. C., Kim, I. K., Kiyohara, Y., Klein, B. E., Klein, R., Kovach, J. L., Kozak, I., Lee, C. J., Lee, K. E., Lichtner, P., Lotery, A. J., Meitinger, T., Mitchell, P., Mohand-Said, S., Moore, A. T., Morgan, D. J., Morrison, M. A., Myers, C. E., Naj, A. C., Nakamura, Y., Okada, Y., Orlin, A., Ortube, M. C., Othman, M. I., Pappas, C., Park, K. H., Pauer, G. J., Peachey, N. S., Poch, O., Priya, R. R., Reynolds, R., Richardson, A. J., Ripp, R., Rudolph, G., Ryu, E., Sahel, J. A., Schaumberg, D. A., Scholl, H. P., Schwartz, S. G., Scott, W. K., Shahid, H., Sigurdsson, H., Silvestri, G., Sivakumaran, T. A., Smith, R. T., Sobrin, L., Souied, E. H., Stambolian, D. E., Stefansson, H., Sturgill-Short, G. M., Takahashi, A., Tosakulwong, N., Truitt, B. J., Tsironi, E. E., Uitterlinden, A. G., van Duijn, C. M., Vijaya, L., Vingerling, J. R., Vithana, E. N., Webster, A. R., Wichmann, H. E., Winkler, T. W., Wong, T. Y., Wright, A. F., Zelenika, D., Zhang, M., Zhao, L., Zhang, K., Klein, M. L., Hageman, G. S., Lathrop, G. M., Stefansson, K., Allikmets, R., Baird, P. N., Gorin, M. B., Wang, J. J., Klaver, C. C., Seddon, J. M., Pericak-Vance, M. A., Iyengar, S. K., Yates, J. R., Swaroop, A., Weber, B. H., Kubo, M., Deangelis, M. M., Leveillard, T., Thorsteinsdottir, U., Haines, J. L., Farrer, L. A., Heid, I. M., and Abecasis, G. R. (2013). Seven new loci associated with age-related macular degeneration. *Nat. Genet.*, 45(4):433–439.

[Fritsche et al., 2014] Fritsche, L. G., Fariss, R. N., Stambolian, D., Abecasis, G. R., Curcio, C. A., and Swaroop, A. (2014). Age-related macular degeneration: genetics and biology coming together. *Annu Rev Genomics Hum Genet*, 15:151–171.

[Gibson, 2011] Gibson, G. (2011). Rare and common variants: twenty arguments. *Nat. Rev. Genet.*, 13(2):135–145.

[Gold et al., 2006] Gold, B., Merriam, J. E., Zernant, J., Hancox, L. S., Taiber, A. J., Gehrs, K., Cramer, K., Neel, J., Bergeron, J., Barile, G. R., Smith, R. T., Hageman, G. S., Dean, M., and Allikmets, R. (2006). Variation in factor B (BF) and complement component 2 (C2) genes is associated with age-related macular degeneration. *Nat. Genet.*, 38(4):458–462.

[Gorin, 2012] Gorin, M. B. (2012). Genetic insights into age-related macular degeneration: controversies addressing risk, causality, and therapeutics. *Mol. Aspects Med.*, 33(4):467–486.

[Goverdhan et al., 2005] Goverdhan, S. V., Howell, M. W., Mullins, R. F., Osmond, C., Hodgkins, P. R., Self, J., Avery, K., and Lotery, A. J. (2005). Association of HLA class I and class II polymorphisms with age-related macular degeneration. *Invest. Ophthalmol. Vis. Sci.*, 46(5):1726–1734.

[Haines et al., 2005] Haines, J. L., Hauser, M. A., Schmidt, S., Scott, W. K., Olson, L. M., Gallins, P., Spencer, K. L., Kwan, S. Y., Noureddine, M., Gilbert, J. R., Schnetz-Boutaud, N., Agarwal, A., Postel, E. A., and Pericak-Vance, M. A. (2005). Complement factor H variant increases the risk of age-related macular degeneration. *Science*, 308(5720):419–421.

[Hirschhorn and Daly, 2005] Hirschhorn, J. N. and Daly, M. J. (2005). Genome-wide association studies for common diseases and complex traits. *Nat. Rev. Genet.*, 6(2):95–108.

[Horie-Inoue and Inoue, 2014] Horie-Inoue, K. and Inoue, S. (2014). Genomic aspects of age-related macular degeneration. *Biochem. Biophys. Res. Commun.*, 452(2):263–275.

[Hughes et al., 2006] Hughes, A. E., Orr, N., Esfandiary, H., Diaz-Torres, M., Goodship, T., and Chakravarthy, U. (2006). A common CFH haplotype, with deletion of CFHR1 and CFHR3, is associated with lower risk of age-related macular degeneration. *Nat. Genet.*, 38(10):1173–1177.

[Idury and Elston, 1997] Idury, R. M. and Elston, R. C. (1997). A faster and more general hidden Markov model algorithm for multipoint likelihood calculations. *Hum. Hered.*, 47(4):197–202.

[Ionita-Laza et al., 2013] Ionita-Laza, I., Lee, S., Makarov, V., Buxbaum, J. D., and Lin, X. (2013). Sequence kernel association tests for the combined effect of rare and common variants. *Am. J. Hum. Genet.*, 92(6):841–853.

[Jakobsdottir et al., 2005] Jakobsdottir, J., Conley, Y. P., Weeks, D. E., Mah, T. S., Ferrell, R. E., and Gorin, M. B. (2005). Susceptibility genes for age-related maculopathy on chromosome 10q26. *Am. J. Hum. Genet.*, 77(3):389–407.

[Kanda et al., 2007] Kanda, A., Chen, W., Othman, M., Branham, K. E., Brooks, M., Khanna, R., He, S., Lyons, R., Abecasis, G. R., and Swaroop, A. (2007). A variant of mitochondrial protein LOC387715/ARMS2, not HTRA1, is strongly associated with age-related macular degeneration. *Proc. Natl. Acad. Sci. U.S.A.*, 104(41):16227–16232.

[Karhunen, 1947] Karhunen, K. (1947). *Üeber lineare Methoden in der Wahrscheinlichkeitsrechnung*. Annales Academiae scientiarum Fennicae. Series A. 1, Mathematica-physica.

[Katta et al., 2009] Katta, S., Kaur, I., and Chakrabarti, S. (2009). The molecular genetic basis of age-related macular degeneration: an overview. *J. Genet.*, 88(4):425–449.

[Kirichenko et al., 2009] Kirichenko, A. V., Belonogova, N. M., Aulchenko, Y. S., and Axenovich, T. I. (2009). PedStr software for cutting large pedigrees for haplotyping, IBD computation and multipoint linkage analysis. *Ann. Hum. Genet.*, 73(Pt 5):527–531.

[Klein et al., 2005] Klein, R. J., Zeiss, C., Chew, E. Y., Tsai, J. Y., Sackler, R. S., Haynes, C., Henning, A. K., SanGiovanni, J. P., Mane, S. M., Mayne, S. T., Bracken, M. B., Ferris, F. L., Ott, J., Barnstable, C., and Hoh, J. (2005). Complement factor H polymorphism in age-related macular degeneration. *Science*, 308(5720):385–389.

[Kruglyak et al., 1995] Kruglyak, L., Daly, M. J., and Lander, E. S. (1995). Rapid multipoint linkage analysis of recessive traits in nuclear families, including homozygosity mapping. *Am. J. Hum. Genet.*, 56(2):519–527.

[Kruglyak et al., 1996] Kruglyak, L., Daly, M. J., Reeve-Daly, M. P., and Lander, E. S. (1996). Parametric and nonparametric linkage analysis: a unified multipoint approach. *Am. J. Hum. Genet.*, 58(6):1347–1363.

[Lander and Kruglyak, 1995] Lander, E. and Kruglyak, L. (1995). Genetic dissection of complex traits: guidelines for interpreting and reporting linkage results. *Nat. Genet.*, 11(3):241–247.

[Lander and Green, 1987] Lander, E. S. and Green, P. (1987). Construction of multilocus genetic linkage maps in humans. *Proc. Natl. Acad. Sci. U.S.A.*, 84(8):2363–2367.

[Lander et al., 2001] Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., Zody, M. C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., Funke, R., Gage, D., Harris, K., Heaford, A., Howland, J., Kann, L., Lehoczky, J., LeVine, R., McEwan, P., McKernan, K., Meldrim, J., Mesirov, J. P., Miranda, C., Morris, W., Naylor, J., Raymond, C., Rosetti, M., Santos, R., Sheridan, A., Sougnez, C., Stange-Thomann, N., Stojanovic, N., Subramanian, A., Wyman, D., Rogers, J., Sulston, J., Ainscough, R., Beck, S., Bentley, D., Burton, J., Clee, C., Carter, N., Coulson, A., Deadman, R., Deloukas, P., Dunham, A., Dunham, I., Durbin, R., French, L., Grafham, D., Gregory, S., Hubbard, T., Humphray, S., Hunt, A., Jones, M., Lloyd, C., McMurray, A., Matthews, L., Mercer, S., Milne, S., Mullikin, J. C., Mungall, A., Plumb, R., Ross, M., Shownkeen, R., Sims, S., Waterston, R. H., Wilson, R. K., Hillier, L. W., McPherson, J. D., Marra, M. A., Mardis, E. R., Fulton, L. A., Chinwalla, A. T., Pepin, K. H., Gish, W. R., Chissoe, S. L., Wendl, M. C., Delehaunty, K. D., Miner, T. L., Delehaunty, A., Kramer, J. B., Cook, L. L., Fulton, R. S., Johnson, D. L., Minx, P. J., Clifton, S. W., Hawkins, T., Branscomb, E., Predki, P., Richardson, P., Wenning, S., Slezak, T., Doggett, N., Cheng, J. F., Olsen, A., Lucas, S., Elkin, C., Uberbacher, E., Frazier, M., Gibbs, R. A., Muzny, D. M., Scherer, S. E., Bouck, J. B., Sodergren, E. J., Worley, K. C., Rives, C. M., Gorrell, J. H., Metzker, M. L., Naylor, S. L., Kucherlapati, R. S., Nelson, D. L., Weinstock, G. M., Sakaki, Y., Fujiyama, A., Hattori, M., Yada, T., Toyoda, A., Itoh, T., Kawagoe, C., Watanabe, H., Totoki, Y., Taylor, T., Weissenbach, J., Heilig, R., Saurin, W., Artiguenave, F., Brottier, P., Bruls, T., Pelletier, E., Robert, C., Wincker, P., Smith, D. R., Doucette-Stamm, L., Rubenfield, M., Weinstock, K., Lee, H. M., Dubois, J., Rosenthal, A., Platzer, M., Nyakatura, G., Taudien, S., Rump, A., Yang, H., Yu, J., Wang, J., Huang, G., Gu, J., Hood, L., Rowen, L., Madan, A., Qin, S., Davis, R. W., Federspiel, N. A., Abola, A. P., Proctor, M. J., Myers, R. M., Schmutz, J., Dickson, M., Grimwood, J., Cox, D. R., Olson, M. V., Kaul, R., Raymond, C., Shimizu, N., Kawasaki, K., Minoshima, S., Evans, G. A., Athanasiou,

M., Schultz, R., Roe, B. A., Chen, F., Pan, H., Ramser, J., Lehrach, H., Reinhardt, R., McCombie, W. R., de la Bastide, M., Dedhia, N., Blocker, H., Hornischer, K., Nordsiek, G., Agarwala, R., Aravind, L., Bailey, J. A., Bateman, A., Batzoglou, S., Birney, E., Bork, P., Brown, D. G., Burge, C. B., Cerutti, L., Chen, H. C., Church, D., Clamp, M., Copley, R. R., Doerks, T., Eddy, S. R., Eichler, E. E., Furey, T. S., Galagan, J., Gilbert, J. G., Harmon, C., Hayashizaki, Y., Haussler, D., Hermjakob, H., Hokamp, K., Jang, W., Johnson, L. S., Jones, T. A., Kasif, S., Kaspryzk, A., Kennedy, S., Kent, W. J., Kitts, P., Koonin, E. V., Korf, I., Kulp, D., Lancet, D., Lowe, T. M., McLysaght, A., Mikkelsen, T., Moran, J. V., Mulder, N., Pollara, V. J., Ponting, C. P., Schuler, G., Schultz, J., Slater, G., Smit, A. F., Stupka, E., Szustakowski, J., Thierry-Mieg, D., Thierry-Mieg, J., Wagner, L., Wallis, J., Wheeler, R., Williams, A., Wolf, Y. I., Wolfe, K. H., Yang, S. P., Yeh, R. F., Collins, F., Guyer, M. S., Peterson, J., Felsenfeld, A., Wetterstrand, K. A., Patrinos, A., Morgan, M. J., de Jong, P., Catanese, J. J., Osoegawa, K., Shizuya, H., Choi, S., Chen, Y. J., and Szustakowki, J. (2001). Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921.

[Larson and Schaid, 2014] Larson, N. B. and Schaid, D. J. (2014). Regularized rare variant enrichment analysis for case-control exome sequencing data. *Genet. Epidemiol.*, 38(2):104–113.

[Laurie et al., 2010] Laurie, C. C., Doheny, K. F., Mirel, D. B., Pugh, E. W., Bierut, L. J., Bhangale, T., Boehm, F., Caporaso, N. E., Cornelis, M. C., Edenberg, H. J., Gabriel, S. B., Harris, E. L., Hu, F. B., Jacobs, K. B., Kraft, P., Landi, M. T., Lumley, T., Manolio, T. A., McHugh, C., Painter, I., Paschall, J., Rice, J. P., Rice, K. M., Zheng, X., and Weir, B. S. (2010). Quality control and quality assurance in genotypic data for genome-wide association studies. *Genet. Epidemiol.*, 34(6):591–602.

[Lee et al., 2014] Lee, S., Abecasis, G. R., Boehnke, M., and Lin, X. (2014). Rare-variant association analysis: study designs and statistical tests. *Am. J. Hum. Genet.*, 95(1):5–23.

[Lee et al., 2012] Lee, S., Wu, M. C., and Lin, X. (2012). Optimal tests for rare variant effects in sequencing association studies. *Biostatistics*, 13(4):762–775.

[Lobo and Shaw, 2008] Lobo, I. and Shaw, K. (2008). Discovery and types of genetic linkage. *Nature education*, 1(1):139.

[Loeve, 1977] Loeve, M. (1977). *Probability Theory I.* Comprehensive Manuals of Surgical Specialties. Springer.

[Luo et al., 2012] Luo, L., Zhu, Y., and Xiong, M. (2012). Quantitative trait locus analysis for next-generation sequencing with the functional linear models. *J. Med. Genet.*, 49(8):513–524.

[Madsen and Browning, 2009] Madsen, B. E. and Browning, S. R. (2009). A groupwise association test for rare mutations using a weighted sum statistic. *PLoS Genet.*, 5(2):e1000384.

[Manolio et al., 2009] Manolio, T. A., Collins, F. S., Cox, N. J., Goldstein, D. B., Hindorff, L. A., Hunter, D. J., McCarthy, M. I., Ramos, E. M., Cardon, L. R., Chakravarti, A., Cho, J. H., Guttmacher, A. E., Kong, A., Kruglyak, L., Mardis, E., Rotimi, C. N., Slatkin, M., Valle, D., Whittemore, A. S., Boehnke, M., Clark, A. G., Eichler, E. E., Gibson, G., Haines, J. L., Mackay, T. F., McCarroll, S. A., and Visscher, P. M. (2009). Finding the missing heritability of complex diseases. *Nature*, 461(7265):747–753.

[Montezuma et al., 2007] Montezuma, S. R., Sobrin, L., and Seddon, J. M. (2007). Review of genetics in age related macular degeneration. *Semin Ophthalmol*, 22(4):229–240.

[Morton, 1955] Morton, N. E. (1955). Sequential tests for the detection of linkage. *Am. J. Hum. Genet.*, 7(3):277–318.

[Müller and Stadtmüller, 2005] Müller, H.-G. and Stadtmüller, U. (2005). Generalized functional linear models. *Annals of Statistics*, pages 774–805.

[Mullins et al., 2000] Mullins, R. F., Russell, S. R., Anderson, D. H., and Hageman, G. S. (2000). Drusen associated with aging and age-related macular degeneration contain proteins common to extracellular deposits associated with atherosclerosis, elastosis, amyloidosis, and dense deposit disease. *FASEB J.*, 14(7):835–846.

[Ott et al., 2011] Ott, J., Kamatani, Y., and Lathrop, M. (2011). Family-based designs for genome-wide association studies. *Nat. Rev. Genet.*, 12(7):465–474.

[Ott et al., 2015] Ott, J., Wang, J., and Leal, S. M. (2015). Genetic linkage analysis in the age of whole-genome sequencing. *Nat. Rev. Genet.*, 16(5):275–284.

[Papachristou et al., 2011] Papachristou, C., Ober, C., and Abney, M. (2011). Genetic variance components estimation for binary traits using multiple related individuals. *Genet. Epidemiol.*, 35(5):291–302.

[Parmeggiani et al., 2012] Parmeggiani, F., Romano, M. R., Costagliola, C., Semeraro, F., Incorvaia, C., D'Angelo, S., Perri, P., De Palma, P., De Nadai, K., and Sebastiani, A. (2012). Mechanism of inflammation in age-related macular degeneration. *Mediators Inflamm.*, 2012:546786.

[Pulst, 1999] Pulst, S. M. (1999). Genetic linkage analysis. *Arch. Neurol.*, 56(6):667–672.

[R Core Team, 2014] R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

[Ramsay and Dalzell, 1991] Ramsay, J. O. and Dalzell, C. (1991). Some tools for functional data analysis. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 539–572.

[Rao, 1948] Rao, C. (1948). Large sample tests of statistical hypotheses concerning several parameters with applications to problems of estimation. *Mathematical Proceedings of the Cambridge Philosophical Society*, 44:50–57.

[Rao, 2005] Rao, C. (2005). Score test: historical review and recent developments. In *Advances in Ranking and Selection, Multiple Comparisons, and Reliability*, pages 3–20. Springer.

[Rivera et al., 2005] Rivera, A., Fisher, S. A., Fritsche, L. G., Keilhauer, C. N., Lichtner, P., Meitinger, T., and Weber, B. H. (2005). Hypothetical LOC387715 is a second major susceptibility gene for age-related macular degeneration, contributing independently of complement factor H to disease risk. *Hum. Mol. Genet.*, 14(21):3227–3236.

[Roberts et al., 2010] Roberts, R., Wells, G. A., Stewart, A. F., Dandona, S., and Chen, L. (2010). The genome-wide association study–a new era for common polygenic disorders. *J Cardiovasc Transl Res*, 3(3):173–182.

[Robinson et al., 2014] Robinson, M. R., Wray, N. R., and Visscher, P. M. (2014). Explaining additional genetic variation in complex traits. *Trends Genet.*, 30(4):124–132.

[Ross, 1996] Ross, S. (1996). *Stochastic processes*. Wiley series in probability and statistics: Probability and statistics. Wiley.

[Schaffner et al., 2005] Schaffner, S. F., Foo, C., Gabriel, S., Reich, D., Daly, M. J., and Altshuler, D. (2005). Calibrating a coalescent simulation of human genome sequence variation. *Genome Res.*, 15(11):1576–1583.

[Schaid et al., 2013] Schaid, D. J., McDonnell, S. K., Sinnwell, J. P., and Thibodeau, S. N. (2013). Multiple genetic variant association testing by collapsing and kernel methods with pedigree or population structured data. *Genet. Epidemiol.*, 37(5):409–418.

[Schifano et al., 2012] Schifano, E. D., Epstein, M. P., Bielak, L. F., Jhun, M. A., Kardia, S. L., Peyser, P. A., and Lin, X. (2012). SNP set association analysis for familial data. *Genet. Epidemiol.*, 36(8):797–810.

[Seddon et al., 2005] Seddon, J. M., George, S., Rosner, B., and Rifai, N. (2005). Progression of age-related macular degeneration: prospective assessment of C-reactive protein, interleukin 6, and other cardiovascular biomarkers. *Arch. Ophthalmol.*, 123(6):774–782.

[Seddon et al., 2009] Seddon, J. M., Reynolds, R., Maller, J., Fagerness, J. A., Daly, M. J., and Rosner, B. (2009). Prediction model for prevalence and incidence of advanced age-related macular degeneration based on genetic, demographic, and environmental variables. *Invest. Ophthalmol. Vis. Sci.*, 50(5):2044–2053.

[Seddon et al., 2003] Seddon, J. M., Santangelo, S. L., Book, K., Chong, S., and Cote, J. (2003). A genomewide scan for age-related macular degeneration provides evidence for linkage to several chromosomal regions. *Am. J. Hum. Genet.*, 73(4):780–790.

[Seddon et al., 2013] Seddon, J. M., Yu, Y., Miller, E. C., Reynolds, R., Tan, P. L., Gowrisankar, S., Goldstein, J. I., Triebwasser, M., Anderson, H. E., Zerbib, J., Kavanagh, D., Souied, E., Katsanis, N., Daly, M. J., Atkinson, J. P., and Raychaudhuri, S. (2013).

Rare variants in CFI, C3 and C9 are associated with high risk of advanced age-related macular degeneration. *Nat. Genet.*, 45(11):1366–1370.

[Slatkin, 2008] Slatkin, M. (2008). Linkage disequilibrium–understanding the evolutionary past and mapping the medical future. *Nat. Rev. Genet.*, 9(6):477–485.

[Stanhope and Abney, 2012] Stanhope, S. A. and Abney, M. (2012). GLOGS: a fast and powerful method for GWAS of binary traits with risk covariates in related populations. *Bioinformatics*, 28(11):1553–1554.

[Strasser et al., 2012] Strasser, D., Neumann, K., Bergmann, H., Marakalala, M. J., Guler, R., Rojowska, A., Hopfner, K. P., Brombacher, F., Urlaub, H., Baier, G., Brown, G. D., Leitges, M., and Ruland, J. (2012). Syk kinase-coupled C-type lectin receptors engage protein kinase C- to elicit Card9 adaptor-mediated innate immunity. *Immunity*, 36(1):32–42.

[Swaroop et al., 2007] Swaroop, A., Branham, K. E., Chen, W., and Abecasis, G. (2007). Genetic susceptibility to age-related macular degeneration: a paradigm for dissecting complex disease traits. *Hum. Mol. Genet.*, 16 Spec No. 2:R174–182.

[Synowiec et al., 2012] Synowiec, E., Pogorzelska, M., Blasiak, J., Szaflik, J., and Szaflik, J. P. (2012). Genetic polymorphism of the iron-regulatory protein-1 and -2 genes in age-related macular degeneration. *Mol. Biol. Rep.*, 39(6):7077–7087.

[Thompson et al., 2007] Thompson, C. L., Jun, G., Klein, B. E., Klein, R., Capriotti, J., Lee, K. E., and Iyengar, S. K. (2007). Genetics of pigment changes and geographic atrophy. *Invest. Ophthalmol. Vis. Sci.*, 48(7):3005–3013.

[Thornton and McPeek, 2007] Thornton, T. and McPeek, M. S. (2007). Case-control association testing with related individuals: a more powerful quasi-likelihood score test. *Am. J. Hum. Genet.*, 81(2):321–337.

[Thornton and McPeek, 2010] Thornton, T. and McPeek, M. S. (2010). ROADTRIPS: case-control association testing with partially or completely unknown population and pedigree structure. *Am. J. Hum. Genet.*, 86(2):172–184.

[Ullah and Finch, 2013] Ullah, S. and Finch, C. F. (2013). Applications of functional data analysis: A systematic review. *BMC Med Res Methodol*, 13:43.

[Visscher et al., 2012] Visscher, P. M., Brown, M. A., McCarthy, M. I., and Yang, J. (2012). Five years of GWAS discovery. *Am. J. Hum. Genet.*, 90(1):7–24.

[Vitart et al., 2010] Vitart, V., Bencić, G., Hayward, C., Skunca Herman, J., Huffman, J., Campbell, S., Bućan, K., Navarro, P., Gunjaca, G., Marin, J., Zgaga, L., Kolcić, I., Polasek, O., Kirin, M., Hastie, N. D., Wilson, J. F., Rudan, I., Campbell, H., Vatavuk, Z., Fleck, B., and Wright, A. (2010). New loci associated with central cornea thickness include COL5A1, AKAP13 and AVGR8. *Hum. Mol. Genet.*, 19(21):4304–4311.

[Wagner, 2013] Wagner, M. J. (2013). Rare-variant genome-wide association studies: a new frontier in genetic analysis of complex traits. *Pharmacogenomics*, 14(4):413–424.

[Wang et al., 2013] Wang, Z., Liu, X., Yang, B. Z., and Gelernter, J. (2013). The role and challenges of exome sequencing in studies of human diseases. *Front Genet*, 4:160.

[Weeks et al., 2000] Weeks, D. E., Conley, Y. P., Mah, T. S., Paul, T. O., Morse, L., Ngo-Chang, J., Dailey, J. P., Ferrell, R. E., and Gorin, M. B. (2000). A full genome scan for age-related maculopathy. *Hum. Mol. Genet.*, 9(9):1329–1349.

[Weeks et al., 2004] Weeks, D. E., Conley, Y. P., Tsai, H. J., Mah, T. S., Schmidt, S., Postel, E. A., Agarwal, A., Haines, J. L., Pericak-Vance, M. A., Rosenfeld, P. J., Paul, T. O., Eller, A. W., Morse, L. S., Dailey, J. P., Ferrell, R. E., and Gorin, M. B. (2004). Age-related maculopathy: a genomewide scan with continued evidence of susceptibility loci within the 1q31, 10q26, and 17q25 regions. *Am. J. Hum. Genet.*, 75(2):174–189.

[Weeks and Lathrop, 1995] Weeks, D. E. and Lathrop, G. M. (1995). Polygenic disease: methods for mapping complex disease traits. *Trends Genet.*, 11(12):513–519.

[Wu et al., 2011] Wu, M. C., Lee, S., Cai, T., Li, Y., Boehnke, M., and Lin, X. (2011). Rare-variant association testing for sequencing data with the sequence kernel association test. *Am. J. Hum. Genet.*, 89(1):82–93.

[Yan et al., 2015] Yan, Q., Tiwari, H. K., Yi, N., Gao, G., Zhang, K., Lin, W. Y., Lou, X. Y., Cui, X., and Liu, N. (2015). A Sequence Kernel Association Test for Dichotomous Traits in Family Samples under a Generalized Linear Mixed Model. *Hum. Hered.*, 79(2):60–68.

[Zareparsi et al., 2005] Zareparsi, S., Buraczynska, M., Branham, K. E., Shah, S., Eng, D., Li, M., Pawar, H., Yashar, B. M., Moroi, S. E., Lichter, P. R., Petty, H. R., Richards, J. E., Abecasis, G. R., Elner, V. M., and Swaroop, A. (2005). Toll-like receptor 4 variant D299G is associated with susceptibility to age-related macular degeneration. *Hum. Mol. Genet.*, 14(11):1449–1455.

[Zhan et al., 2013] Zhan, X., Larson, D. E., Wang, C., Koboldt, D. C., Sergeev, Y. V., Fulton, R. S., Fulton, L. L., Fronick, C. C., Branham, K. E., Bragg-Gresham, J., Jun, G., Hu, Y., Kang, H. M., Liu, D., Othman, M., Brooks, M., Ratnapriya, R., Boleda, A., Grassmann, F., von Strachwitz, C., Olson, L. M., Buitendijk, G. H., Hofman, A., van Duijn, C. M., Cipriani, V., Moore, A. T., Shahid, H., Jiang, Y., Conley, Y. P., Morgan, D. J., Kim, I. K., Johnson, M. P., Cantsilieris, S., Richardson, A. J., Guymer, R. H., Luo, H., Ouyang, H., Licht, C., Pluthero, F. G., Zhang, M. M., Zhang, K., Baird, P. N., Blangero, J., Klein, M. L., Farrer, L. A., DeAngelis, M. M., Weeks, D. E., Gorin, M. B., Yates, J. R., Klaver, C. C., Pericak-Vance, M. A., Haines, J. L., Weber, B. H., Wilson, R. K., Heckenlively, J. R., Chew, E. Y., Stambolian, D., Mardis, E. R., Swaroop, A., and Abecasis, G. R. (2013). Identification of a rare coding variant in complement 3 associated with age-related macular degeneration. *Nat. Genet.*, 45(11):1375–1379.

[Ziegler et al., 2010] Ziegler, A., König, I. R., and Pahlke, F. (2010). *A Statistical Approach to Genetic Epidemiology: Concepts and Applications, with an e-learning platform*. John Wiley & Sons.